JBoss Operations Network 3.1 How to Manage JBoss Servers with JBoss ON

managing JBoss applications
Edition 3.1.2

Landmann

JBoss Operations Network 3.1 How to Manage JBoss Servers with JBoss ON

managing JBoss applications
Edition 3.1.2

Landmann
rlandmann@redhat.com

Legal Notice

Copyright © 2013 Red Hat, Inc..

This document is licensed by Red Hat under the Creative Commons Attribution-ShareAlike 3.0 Unported License. If you distribute this document, or a modified version of it, you must provide attribution to Red Hat, Inc. and provide a link to the original. If the document is modified, all Red Hat trademarks must be removed.

Red Hat, as the licensor of this document, waives the right to enforce, and agrees not to assert, Section 4d of CC-BY-SA to the fullest extent permitted by applicable law.

Red Hat, Red Hat Enterprise Linux, the Shadowman logo, JBoss, MetaMatrix, Fedora, the Infinity Logo, and RHCE are trademarks of Red Hat, Inc., registered in the United States and other countries.

Linux ® is the registered trademark of Linus Torvalds in the United States and other countries.

Java ® is a registered trademark of Oracle and/or its affiliates.

XFS ® is a trademark of Silicon Graphics International Corp. or its subsidiaries in the United States and/or other countries.

MySQL ® is a registered trademark of MySQL AB in the United States, the European Union and other countries.

Node.js ® is an official trademark of Joyent. Red Hat Software Collections is not formally related to or endorsed by the official Joyent Node.js open source or commercial project.

The OpenStack ® Word Mark and OpenStack Logo are either registered trademarks/service marks or trademarks/service marks of the OpenStack Foundation, in the United States and other countries and are used with the OpenStack Foundation's permission. We are not affiliated with, endorsed or sponsored by the OpenStack Foundation, or the OpenStack community.

All other trademarks are the property of their respective owners.

Abstract

Once JBoss Operations Network is installed, you are ready to set up your inventory. The inventory in JBoss ON lists all of the recognized platforms, servers, and services

Table of Contents

Document Information **5**
1. Giving Feedback 5
2. Document History 5

Chapter 1. How JBoss ON Manages JBoss Resources **6**
1.1. How JBoss ON Works with JBoss Software 6
1.2. What's Covered in This Guide 6
1.3. Installing JBoss Plug-in Packs 6

Part I. General Tasks **8**

Chapter 2. Setting up a Custom JVM for Discovery **9**
2.1. Required JVM Configuration for Discovery 9
2.2. Excluding Java Processes from Discovery 9
2.3. Manually Importing a JVM Resource 9

Chapter 3. Enabling the Agent to Connect to Secured JMX Servers **11**

Part II. Managing JBoss EAP 6 (AS 7) **12**

Chapter 4. The Structure of JBoss EAP 6 ..... **13**
4.1. "Classic" Structure: Standalone Servers 13
4.2. Separating Configuration and Real-Time Operations: Domains 14
4.3. EAP 6 Resources in JBoss ON 17
4.4. The Purpose of Managing EAP 6 Resources with JBoss ON 17
4.5. Topics Covered in This Guide 17

Chapter 5. Upgrading the JBoss EAP 6 Resource Plug-in **19**

Chapter 6. Setting up JBoss EAP 6 Instances **21**
6.1. Configuration for Servers and Profiles 21
6.1.1. Differences for Standalone Servers and Domains 21
6.1.2. Requried Management Interfaces on EAP 6 21
6.1.3. Configuration Features in JBoss ON 22
6.2. Creating Management Users 22
6.2.1. Creating a Management User Through JBoss ON 22
6.2.2. Creating a Management User in EAP 6 23
6.3. Creating a Dynamic Group for EAP 6 Resources 23
6.4. Setting Start Script Arguments, Environment Variables, and JAVA_OPTS 24
6.4.1. Start Script Discovery and Settings 24
6.4.2. Start Script Arguments and Drift Monitoring 25
6.4.3. Changing Start Script Configuration 25
6.5. Changing Port Numbers 26
6.5.1. Changing Socket Binding Ports 26
6.5.2. Changing Port Offsets for Server Groups in a Domain 27
6.6. Editing Network Interfaces 28
6.7. Setting System Properties 29
6.8. Adding System Paths 30
6.9. Editing Connection Settings 31
6.9.1. Changing the General Properties for an EAP 6 Server 31
6.9.2. Viewing Installation Paths for EAP 6 Child Resources 32
6.10. Viewing Installed Extensions 33
6.11. Reloading the Server Configuration 34
6.12. Controlling Configuration Drift 34

6.13. Tracking and Reverting Configuration Changes 35

Chapter 7. Creating JBoss EAP 6 Resources **37**
7.1. Tracking the Child History 37
7.2. Creating Server Groups 37
7.3. Creating Managed Servers 37
7.4. Changing JVM Definitions 38
7.4.1. JVM Definitions as Resources 39
7.4.2. Creating a JVM Definition 39
7.4.3. Editing a JVM Definition 41
7.5. A Short List of Parent-Child Resources 41

Chapter 8. Deploying Web Applications **43**
8.1. Runtime Information and Deployment Resources 43
8.1.1. Views of Deployments 43
8.1.2. Deployment Paths for Standalone Servers and Domains 44
8.2. Deploying Web Applications to a Domain 44
8.3. Assigning Web Applications to a Server Group 46
8.4. Extended Example: Assigning Web Applications and Managing Updates 48
8.5. Enabling and Disabling Web Applications 48
8.6. Updating Deployment Content 48
8.7. Deploying Web Applications to a Standalone Server 50
8.7.1. Deploying a Web Application as a Child Resource 50
8.7.2. Deploying Web Applications Through Bundles 51
8.8. Tracking Content History and Reverting Changes 51
8.9. Troubleshooting Deployments 52

Chapter 9. Monitoring JBoss EAP 6 Resources **54**
9.1. Runtime Information and JBoss ON Monitoring 54
9.2. Configuring Response Time Metrics for JBoss EAP 6/AS 7 55
9.2.1. Installing the Response Time Filters 55
9.2.2. Enabling the Call-Time Metric 60
9.3. Setting up Monitoring for EAP 6 Resources 61
9.3.1. Enabling Additional Metrics 62
9.3.2. Availability Monitoring 62
9.4. Configuring Events Monitoring 64
9.5. Alerting on JBoss EAP 6 Resources 65

Chapter 10. Using the mod_cluster Services in EAP 6 **66**
10.1. About mod_cluster and JBoss ON 66
10.2. Configuring Multicast for Load Balancing 66
10.3. Excluding Web Contexts from Discovery 68
10.4. Configuring Web Context Metrics 70

Part III. Managing JBoss EAP 5 **72**

Chapter 11. Discovering JBoss EAP/AS 5 Servers **73**
11.1. Discovering and Managing the JBoss AS/EAP 5 JVM 73
11.2. Enabling Remote Access to JMX and Profile Service 73
11.3. Setting Start Script Arguments, Environment Variables, and JAVA_OPTS 74
11.3.1. Start Script Discovery and Settings 74
11.3.2. Start Script Arguments and Drift Monitoring 74
11.3.3. Changing Start Script Configuration 75

Chapter 12. Creating JBoss EAP 5 Child Resources **77**
12.1. Creating Data Sources 77
12.2. Creating Connection Factories 77

12.3. Creating JMS Queues and Topics 77

Chapter 13. Deploying Applications **79**
13.1. Setting Permissions for Agent and Resource Users 79
13.2. Deploying EAR and WAR Files 79
13.3. Updating Applications 81
13.4. Deleting an Application 83

Chapter 14. Applying JBoss Patches from the Patch RSS Feed **85**
14.1. Planning How to Patch JBoss Resources 85
14.1.1. Identifying Content Sources 85
14.1.2. Planning Manual Steps 85
14.2. Enabling the Default JBoss Patch Content Source 85
14.3. Subscribing a Specific Resource to the Default JBoss Patch Repository 87
14.4. Subscribing Multiple JBoss Resources to the Default JBoss Patch Repository 87
14.5. Applying a Patch 88

Chapter 15. Configuring HTTP Response Time Metrics for JBoss EAP/AS 5 **90**
15.1. Installing the Response Time Filters 90
15.2. Configuring HTTP Response Time Metrics 93

Chapter 16. Managing mod_cluster Deployments for JBoss EAP 5 (Tech Preview) **95**
16.1. About mod_cluster 95
16.2. Managing mod_cluster 96
16.3. Managing Web Applications Contexts 97

Interactions with System Users for Agents and Resources **99**
A.1. The Agent User 99
A.2. Agent Users and Discovery 99
A.3. Users and Management Tasks 99
A.4. Using sudo with JBoss ON Operations 100

Index **100**
D 100
J 101
M 101
P 101
R 101
T 102

Document Information

While JBoss Operations Network can monitor, manage, configure, and deploy content an a wide variety of applications and systems, it has a special relationship with JBoss products.

This guide provides additional layers of details for common management tasks and it provides information about special use cases and operations available with different JBoss resources.

For more general management information, see the rest of theJBoss Operations Network administrative documentation at http://docs.redhat.com/docs/en-US/JBoss_Operations_Network/index.html.

1. Giving Feedback

If there is any error in this guide or there is any way to improve the documentation, please let us know. Bugs can be filed in Bugzilla, http://bugzilla.redhat.com/bugzilla. Make the bug report as specific as possible, so we can be more effective in correcting any issues:

1. Select the **JBoss** products group.
2. Select **JBoss Operations Network** from the list.
3. Set the component to **Documentation**.
4. Set the version number to 3.1.2.
5. For errors, give the page number (for the PDF) or URL (for the HTML), and give a succinct description of the problem, such as incorrect procedure or typo.

   For enhancements, put in what information needs to be added and why.
6. Give a clear title for the bug. For example, **"Incorrect command example for setup script options"** is better than **"Bad example"**.

We appreciate receiving any feedback — requests for new sections, corrections, improvements, enhancements, even new ways of delivering the documentation or new styles of docs.

2. Document History

Revision 3.1.2-1.400	**2013-10-31**	**Rüdiger Landmann**
Rebuild with publican 4.0.0		
Revision 3.1.2-1	**January 23, 2013**	**Ella Deon Lackey**
Updated 'deploy to server-group' operation to be renamed and to clarify the process for promoting content in a domain, Bugzilla 886167. Updated response time configuration subsystem module name, Bugzilla 870228. Added a reference that JON requires the HTTP management interface in order to discover EAP 6 resources, Bugzilla 847280. Fixing typos.		
Revision 3.1.1-1	**September 19, 2012**	**Ella Deon Lackey**
Bug fixing for JBoss Operations Network 3.1.1.		
Revision 3.1-1	**June 12, 2012**	**Ella Deon Lackey**
The initial release of JBoss Operations Network 3.1.		

Chapter 1. How JBoss ON Manages JBoss Resources

JBoss Operations Network provides extra tools that help manage JBoss server instances. These management tools cover everything from manually configuring a JBoss inventory to applying JBoss patches.

1.1. How JBoss ON Works with JBoss Software

JBoss AS/EAP is an application server, so its core goal is delivering web content. In order to manage JBoss applications effectively, JBoss ON manages both the application server itself and the content that it delivers.

At the server level, JBoss ON treats JBoss servers and services as resources, just like other types of managed resources. Each major JBoss server type — like EAP, SOA-P, Data Services, or Business Rules Management — is implemented through an agent resource plug-in; these plug-ins are available in separate, independently-installed *plug-in packs*. These plug-ins define a variety of related services and projects that support the main server type. For example, the EAP plug-in pack includes agent plug-ins for EAP, Hibernate, Cache Service, JMS, and JMX.

JBoss ON provides support for application servers and services by managing and monitoring:

- Monitoring, event logging, availability, and alerting ("Setting up Monitoring, Alerts, and Operations")
- Managing software updates and patches (Chapter 14, *Applying JBoss Patches from the Patch RSS Feed*)

In JBoss ON, there is a very close relationship between JBoss resources and JBoss content. The web content — like EARs, WARs, and web contexts — is treated as a hybrid kind of resource. They have a defined parent-child hierarchy and have management tasks like starting and stopping instances, metrics and alerting, and configurtion properties as do other types of resources.

But content-backed resources are simultaneously treated as software packages, with update histories, content repositories, and the ability to revert to previous versions.

Managing JBoss content resources, then, focuses on the relationship of the content to the application server:

- Deployed web applications (Chapter 13, *Deploying Applications*)
- Clusters and web contexts (Chapter 16, *Managing mod_cluster Deployments for JBoss EAP 5 (Tech Preview)*)

The key for management (of resources and content) is that everything is unified into a central location, across JBoss applications, across domains, and across machines.

1.2. What's Covered in This Guide

In a very general sense, JBoss ON handles JBoss resources the same as any other: it provides monitoring and alerting, manages configuration and drift, and displays and controls the inventory.

On another level, because of the tight integration between JBoss products and JBoss ON, JBoss ON provides unique ways to manage JBoss servers. JBoss ON can streamline common operations in ways that are easier than trying to do it directly on JBoss resources. It is also centralized and offers a focused view on JBoss resources.

So, the purpose of this guide is not to delve into how to use JBoss ON generally. This guide is focused on how to use JBoss ON to perform specific tasks with JBoss resources and, therefore, how to manage JBoss resources more effectively. This includes:

- Content management for EARs and WARs and other web applications
- Managing JBoss server clusters
- Managing JBoss server domains
- Monitoring JBoss instances

For concepts and general procedures for using JBoss ON, see the full documentation set at http://docs.redhat.com/docs/en-US/JBoss_Operations_Network/index.html.

1.3. Installing JBoss Plug-in Packs

All resources are identified to the server and the agent through *agent plug-ins*. Agent plug-ins defined support resources, including supported metrics, operations, configuration propertities, and resource versions.

JBoss products are supported through their own special plug-ins which are available in separate *plug-in packs*. These plug-ins packs have to be installed before the JBoss products can be discovered and managed.

Table 1.1. Supported JBoss Products

Product	Versions	Plug-in Pack
JBoss AS or EAP	4.x 5.x	EAP
JBoss AS	7.1	Tech Preview
mod_cluster	1.1.2	Tech Preview
Hibernate		EAP
JBoss Cache Service	1 3	EAP
JMX		EAP
JMS Manager Service (HornetQ)		EAP
Business Rules Management Service (BRMS/Drools)	5.x	BRMS
Data Services Platform (EDS-P/Teiid)	5.x	EDS
Developer Studio (ModeShape)	4.x	ModeShape
Tomcat (Web Services)	1.x	EWS
ESB Services (SOA-P)	4.x 5.x	SOA-P

To install JBoss plug-ins:

1. Download the plug-in JAR files from the Customer Support Portal.
   In the Customer Support Portal, click **Software**, and then select the **JBoss ON for** *Plug-in* drop-down box.
2. Download the plug-in packs.
3. Unzip the additional plug-in packs. This creates a subdirectory with the name **jon-plugin-pack-*plugin_name*-3.1.2.GA1**.
4. Copy the new plug-ins from the **jon-plugin-pack-*plugin_name*-3.1.2.GA1/** directory to the JBoss ON server plug-in directory.

   ```
   [root@server rhq-agent]# cp myDirectory/jon-plugin-pack-plugin_name-
   3.1.2.GA1/* /opt/jon/jon-server-3.1.2.GA1/plugins
   ```

   The server polls this directory every few minutes to look for updated plug-ins.
5. Once the plug-ins have been deployed to the server, reload the agent plug-ins on the agents themselves.
   This can be done from the command line using the agent's **plugins** command:

   ```
   > plugins update
   ```

   This can also be done in the JBoss ON GUI by scheduling an *update plugins* operation for an agent or a group or agents.

Part I. General Tasks

Chapter 2. Setting up a Custom JVM for Discovery

The Generic JMX Plug-in detects Java processes. This is a very simple plug-in; aside from the ability to exclude some types of JVMs, the plug-in detects *any* properly-configured Java process and imports it is as a JMX server. For a generic JMX server, all of its subsystems — logging, threads, memory, and others — are also imported as children of the JMX server.

All of this background information is covered in the JMX resource documentation in the *Resource Reference: Monitoring, Operation, and Configuration Options*. Writing a custom plugin is covered in *Writing Custom Plug-ins*.

A JVM has to be configured in a certain way for JBoss ON to be able to discover it and to add it to the inventory.

2.1. Required JVM Configuration for Discovery

The agent discovers and subsequently manages a resource by identifying the resource based on certain parameters and then connecting to the agent over those discovered settings. For a Java process to be discovered using the Generic JMX Plug-in, it has to be configured to be connected to in one of two ways:

- Sun JMX remoting is enabled, with a port system property specified in the command line.

```
-Dcom.sun.management.jmxremote.port=12345 com.xyz.MyAppMain
```

- A Sun/Oracle-compatible Java process is accessible through the **`com.sun.tools.attach`** API, and the resource key is specified as a system property in the command line.

```
-Dorg.rhq.resourceKey=KEY com.xyz.MyAppMain
```

2.2. Excluding Java Processes from Discovery

The Generic JMX Plug-in is designed to be extended so that custom plug-ins can be based off it to detect and manage specific types of Java processes. For example, the JBoss ON agent, as a Java process, would normally be detected by the Generic JMX Plug-in, but it is discovered by the Agent Plug-in, which trumps the Generic JMX Plug-in. Similarly, JBoss EAP and Tomcat servers are also discovered by server-specific plug-ins, not the generic plug-in.

Resources which can be discovered by another plug-in should be excluded from the Generic JMX Server discovery, or there could be (spurious) conflict errors recorded in the agent log.

The Java processes to exclude from the discovery scan are defined in the JBoss ON agent configuration. There is a Java option for the agent, **`rhq.jmxplugin.process-filters`**, which lists strings to ignore specifically from the Generic JMX Plug-in discovery scan. If a Java process contains any of the strings in the filter, it is excluded from discovery as a JMX server [1].

The default agent configuration filters out the classes for the agent, the JBoss ON server, and Tomcat servers:

```
RHQ_AGENT_ADDITIONAL_JAVA_OPTS="-Drhq.jmxplugin.process-
filters=org.rhq.enterprise.agent.AgentMain,org.jboss.Main,catalina.startup.Bootstrap
"
```

To add excludes filters:

1. Open the agent configuration file.

```
[jbosson-agent@server ~]$ vim agentRoot/rhq-agent/conf/agent-
configuration.xml
```

2. In the **`RHQ_AGENT_ADDITIONAL_JAVA_OPTS`** link, add the string to exclude to the **`rhq.jmxplugin.process-filters`** option. This can be the classname or any other identifying string which is in the command line for the given process.

   For example:

```
RHQ_AGENT_ADDITIONAL_JAVA_OPTS="-Drhq.jmxplugin.process-
filters=org.rhq.enterprise.agent.AgentMain,org.jboss.Main,catalina.startup.Boot
strap,com.abc.OtherAppMain"
```

   The **`rhq.jmxplugin.process-filters`** value is a comma-separated list of strings.

3. Restart the agent with the **`--config`** option to load the new configuration.

```
[jbosson-agent@server ~]$ agentRoot/rhq-agent/bin/rhq-agent.sh --config
```

2.3. Manually Importing a JVM Resource

As Section 2.1, “Required JVM Configuration for Discovery” describes, there are only two connection configurations that can be used for a Java process for it to be discovered automatically by the agent.

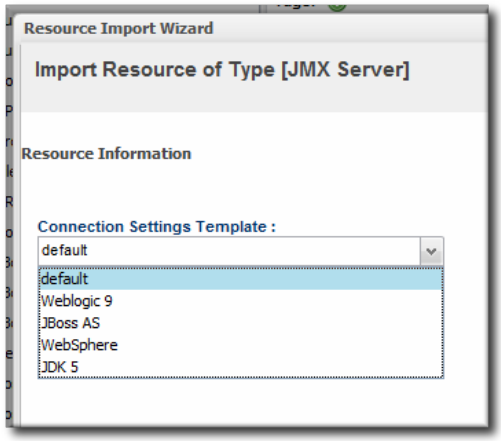
Specifically, those two connection settings both use a Sun management API, for remoting or for attach.

Any Java process can be imported into the inventory manually, even if it does not use the expected configuration for autodiscovery, as long as it enables JMX remoting in a supported form, meaning Sun or IBM remoting.

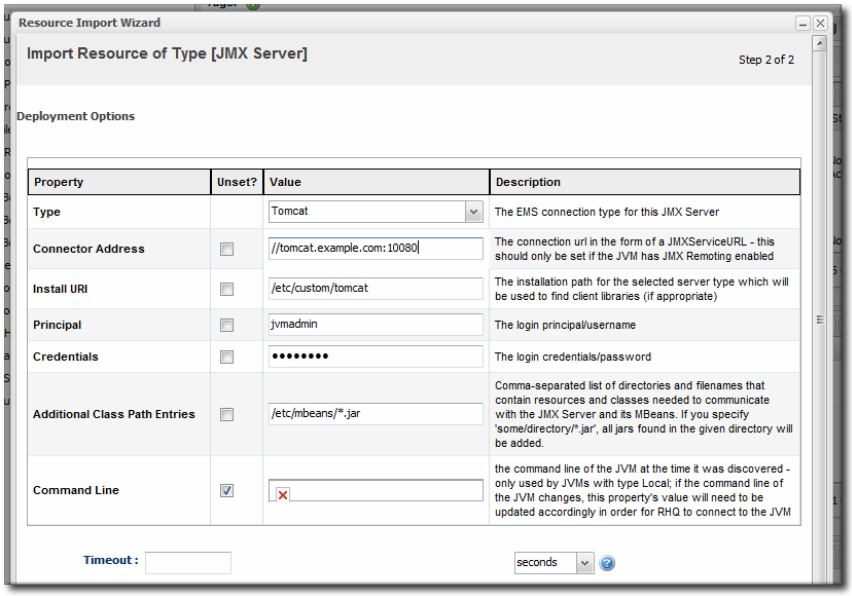
NOTE

The JVM instance has to be running for the resource to be discovered and imported.

1. Click the **Inventory** tab in the top menu.
2. Select the platform resource.
3. Click the **Inventory** tab of the platform.
4. Click the **Import** button in the bottom of the **Inventory** tab, and select the JMX server resource type.
5. Select the type of JVM, and set all of the connection properties correctly, depending on the type of JVM selected.



6. Fill in the connection information for the JVM. This varies depending on the JVM type, but it includes options like a URL and port, directory paths for client libraries, directory paths for classes, and login credentials.


Resource Import Wizard
Import Resource of Type [JMX Server]
Step 2 of 2
Deployment Options
Property
Unset?
Value
Description
Type
Tomcat
The EMS connection type for this JMX Server
Connector Address
//tomcat.example.com:10080
The connection url in the form of a JMXServiceURL - this should only be set if the JVM has JMX Remoting enabled
Install URI
/etc/custom/tomcat
The installation path for the selected server type which will be used to find client libraries (if appropriate)
Principal
jvmadmin
The login principal/username
Credentials
••••••••
The login credentials/password
Additional Class Path Entries
/etc/mbeans/*.jar
Comma-separated list of directories and filenames that contain resources and classes needed to communicate with the JMX Server and its MBeans. If you specify 'some/directory/*.jar', all jars found in the given directory will be added.
Command Line
the command line of the JVM at the time it was discovered - only used by JVMs with type Local; if the command line of the JVM changes, this property's value will need to be updated accordingly in order for RHQ to connect to the JVM
Timeout :
seconds

7. Click the **Finish** to import the instance.

[1] The excluded process can still be discovered by another agent plug-in.

Chapter 3. Enabling the Agent to Connect to Secured JMX Servers

By default, JBoss EAP has its JMX server running in secure mode. However, while the agent can *discover* a JMX server in secure mode, it cannot *connect* to that secured JMX server because it cannot detect the proper credentials.

For example, the JMX server has these system properties:

```
-Dcom.sun.management.jmxremote
-Dcom.sun.management.jmxremote.port=5222
-Dcom.sun.management.jmxremote.ssl=false
-Dcom.sun.management.jmxremote.authenticate=true
-Dcom.sun.management.jmxremote.password.file=/jmxremote.password
-Dcom.sun.management.jmxremote.access.file=/jmxremote.access
```

The agent's JMX plug-in examines the command line for the JMX server's process. It detects the port to use to connect to the JMX server, but it cannot read the password and access files to get the JMX server's credentials.

NOTE

Because the agent cannot connect to the JMX server, it assigns the resource a DOWN availability state, even if the server is running fine.

There are several ways to enable the agent to connect to the JMX server.

Edit the jmx-console-users.properties File

The agent generally reads the connection credentials from the **`jmx-console-*.properties`** file in the *JbossASInstallDir***`/server/default/conf/props/`** directory.

When the JMX server is running secured, there are no entries in the **`jmx-console-users.properties`** file, so there is no way for the agent to get the credentials.

1. Open the **`jmx-console-*.properties`** file for editing. For example:

   ```
   [root@server ~]# vim JbossASInstallDir/server/default/conf/props/jmx-
   console-users.properties
   ```

2. Uncomment or add a line for the admin user.

   ```
   admin=admin
   ```

If that does not work, then edit the connection settings for the resource.

Edit the Connecting Settings to Use the Remote Access Files

By default, the agent uses the **`jmx-console-*.properties`** file for a username, not the access files. It is possible to change the connection settings for the resource so that the agent uses the access files, going through the remote endpoint, which were specified in the JMX server's command line.

1. Click the **`Inventory`** tab in the top menu.
2. Search for the JMX server in the **`Servers`** area of the **`Inventory`**, or open the JBoss EAP instance and navigate through its children to find the JMX server instance.
3. On the JMX server's entry page, open the **`Inventory`** tab, and select the **`Connection Settings`** subtab.
4. Enter the user name and password to set in the JMX remote access files.
5. Click the **`Save`** button.

Edit the Connection Settings to Connect Through the Parent Resource

JBoss ON can connect to the parent resource, and then use that to connect to the JMX server, rather than connecting through the remoting endpoint. This does not require using any user credentials, since the parent can connect to the child resource using internal authentication.

1. Click the **`Inventory`** tab in the top menu.
2. Search for the JMX server in the **`Servers`** area of the **`Inventory`**, or open the JBoss EAP instance and navigate through its children to find the JMX server instance.
3. On the JMX server's entry page, open the **`Inventory`** tab, and select the **`Connection Settings`** subtab.
4. Unset all of the connection properties *except* for the **`Type`** property.
5. For the **`Type`** property, select the **`Parent`** value.
6. Click the **`Save`** button.

Part II. Managing JBoss EAP 6 (AS 7)

Chapter 4. The Structure of JBoss EAP 6

The key features of JBoss EAP 6 are a modular class-loading structure and a domain framework which centralizes configuration in a single controller. These two features have one big asset in common: flexibility. The idea is to start with a small, lightweight application server and then add in the modules for the desired functionality and the server instances for a naturally distributed system.

JBoss ON works within the changeability of that system to bring clarity into how those modular, distributed resources are related and to provide an extra skin of both administrative control and information visibility.

JBoss ON imposes a hierarchy and structure on JBoss EAP 6 installation, but before you can have an idea of how that structure is applied, it helps to have an understanding of the order that EAP 6 itself has.

NOTE

JBoss EAP 6 topologies for standalone and domain servers are described in the JBoss AS 7 project documentation.

4.1. "Classic" Structure: Standalone Servers

Most of the functionality for the JBoss EAP 6 server is implemented through independent classes or *subsystems*. Which subsystems a server uses, and their configuration properties or configured instances, are defined in a *profile*.

A standalone instance for EAP 6, much as in EAP 5, is a collection of those defined subsystems, a set of socket bindings to define communications, network interfaces, and other settings.

The entire server instance is defined in its **`standalone.xml`** file. Almost every entry in that file is identified as a child of the server. For example, these subsystem entries create the **`ee`** and **`jmx`** child resources for the server:

```
<subsystem xmlns="urn:jboss:domain:ee:1.1"/>
<subsystem xmlns="urn:jboss:domain:jmx:1.1">
    <show-model value="true"/>
    <remoting-connector/>
</subsystem>
```

Likewise, socket bindings, network interfaces, and deployed content, which are defined in the configuration file, are discovered as resources.

Subsystems like datasources and logging configuration can have multiple instances defined; these are all configured as child resources of the subsystem. Because of the modular nature of the EAP 6 server, the exact subsystems — and therefore the exact children in JBoss ON — vary with the subsystem and other definitions in the configuration file.

Overall, because of the number of subsystems and other configuration settings, the standalone server has a very wide, flat, and shallow inventory tree. JBoss ON imposes little additional hierarchy on the standalone server structure. It reflects the way the server is configured in **`standalone.xml`**.

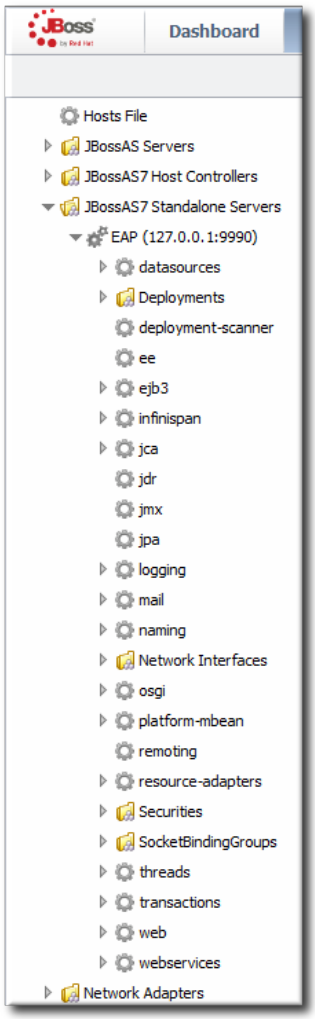


Figure 4.1. Standalone Server Configuration

Management and operations for the standalone server are self-contained. **Every child node for the standalone server combines both its configuration and its runtime environment (monitoring, alerting, and operations) in the same location. To change the configuration for a datasource, for example, edit the datasource entry directly.** Every standalone server is managed independently of any other server, even in the same cluster or the same machine.

All of this is comparable to the structure and functionality of a JBoss EAP 5 server, with some slight differences in the organization of the inventory.

4.2. Separating Configuration and Real-Time Operations: Domains

The structure of the domain is entirely different, in a way that embraces a more modular design.

A domain divides the application server into two conceptual halves: server configuration and runtime operations.

Configuration is centralized into the domain controller. The domain controller defines the profile and subsystem configurations, JVM settings, interfaces, and other settings in a single place.

The functions of the application server are carried out by *managed* servers, which can be installed on multiple machines. These managed servers do not define their own configuration (with a handful of exceptions); they accept the profile structures from the domain and serve the web applications deployed through the domain.

There is a secondary level of organization domains and managed servers called *server groups*. Server groups are defined as part of the domain configuration. They create environments for managed servers; they are configuration nodes, in a sense. A domain can have multiple profiles, JVM definitions, and web applications; it is a central repository of all possible configuration. Only specific configuration and content is assigned to a server group, and that specific configuration is what is applied to the managed servers.

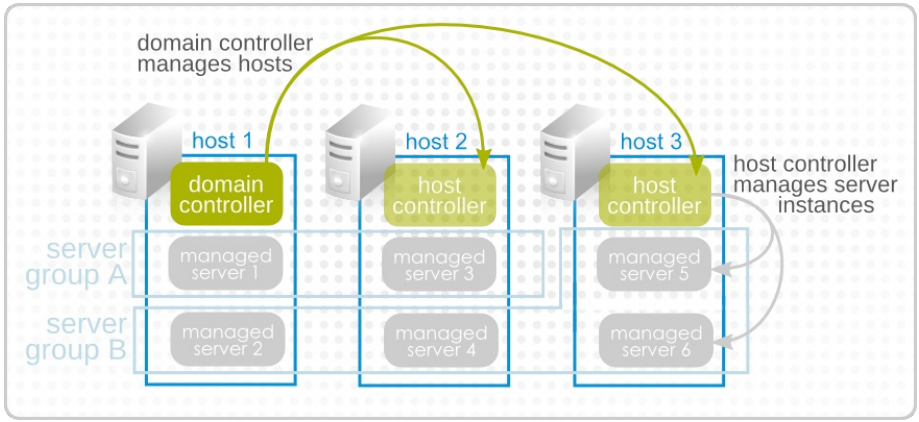


Figure 4.2. Simple Domain Structure

The domain configuration is defined in the **`domain.xml`** file. This lists all of the configured profiles and subsystems, server groups, socket binding configuration, system properties, deployments, and other settings. As with the standalone server, almost every entry is discovered and added to the inventory as a resource. For example, this creates a server group resource, with a child deployment resource and a child JVM resource for the server group.

Example 4.1. Server Group domain.xml Entry

```
        <server-group name="main-server-group" profile="full">
            <jvm name="default">
                <heap size="1303m" max-size="1303m"/>
                <permgen max-size="256m"/>
            </jvm>
            <socket-binding-group ref="full-sockets"/>
            <deployments>
                <deployment name="sample2.war" runtime-name="sample2.war"/>
            </deployments>
 </server-group>
```

The instances that carry out the operations are identified in the **`host.xml`** file. These managed servers have virtually no configuration of their own; they simply point back to the original server group configuration to use in the domain.

```
    <servers>
        <server name="server-one" group="main-server-group"/>
        <server name="server-two" group="other-server-group">
            <!-- server-two avoids port conflicts by incrementing the ports in
                 the default socket-group declared in the server-group -->
            <socket-bindings port-offset="150"/>
        </server>
    </servers>
```

The domain, through its profiles and its server groups, defines the overall configuration. The managed servers are workers under that configuration. This division between configuration and operation, particularly the emphasis on the functionality of the domain as a whole rather than any individual member, is reflected in the EAP 6 management console. The **`Profile`** and **`Servers`** areas manage domain configuration, while **`Runtime`** displays current statistics and deployed web applications. The focus is always on domain configuration and domain information.

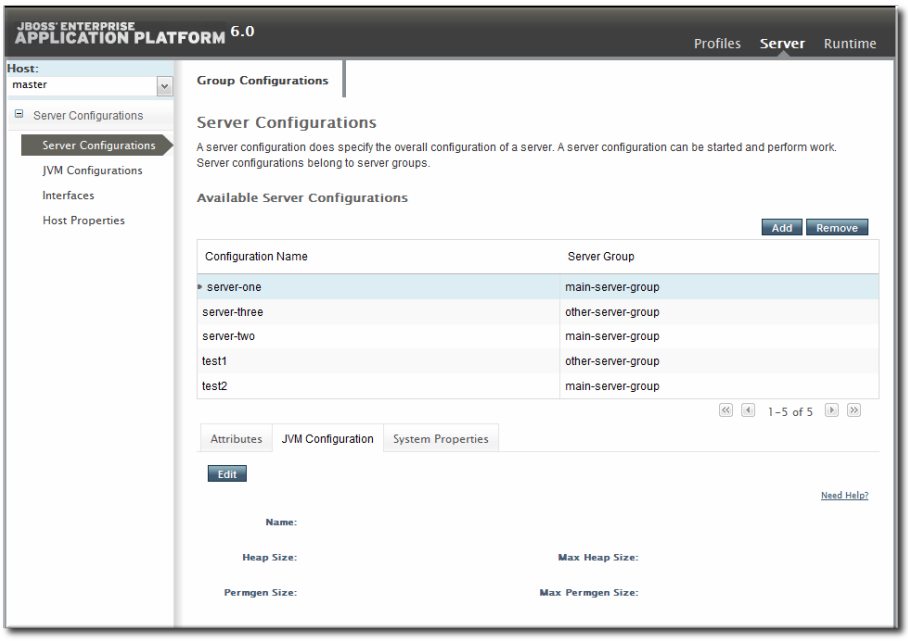


Figure 4.3. EAP 6 Console

Managing domain resources in JBoss ON is less about managing the configuration than it is managing information. One of the strengths of the JBoss ON inventory structure is that is exposes the components of the domain very clearly and helps delineate the relationships between those resources.

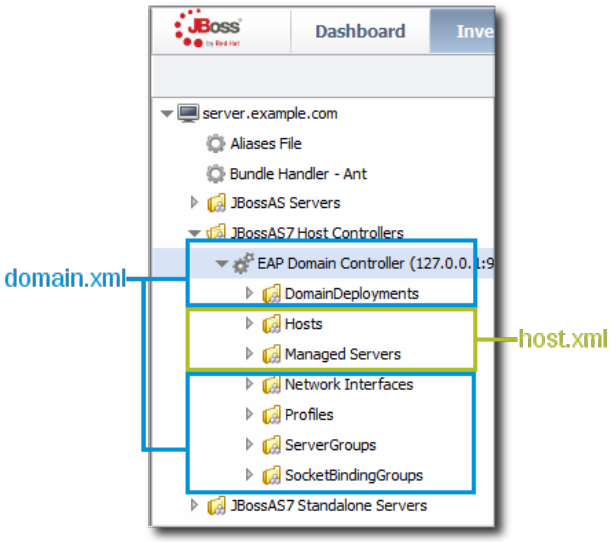


Figure 4.4. Domain Resources

For example, the entry for a managed server lists all of the subsystems defined in its server group's profile, it lists its host controller definition, and it lists managed web applications deployed on it (through the server group). This consolidates all of the configuration information that is applied to that one worker instance, which makes the monitoring information on that managed server more valuable.

There is a shade of artificiality in the JBoss ON inventory in order to maintain the domain relationships. The domain is a single, overarching entity, but the components of the domain can span multiple platforms and resources on those platforms. The domain controller is considered the parent resource for all domain resources, regardless of their platform. This is hinted at in Figure 4.5, “Domain Components in the JBoss ON Inventory”. The domain controller is on Platform 1, so even though a host controller and two managed servers are on Platform 2, the host controller and managed resources are shown as children of the domain resource (on Platform 1).

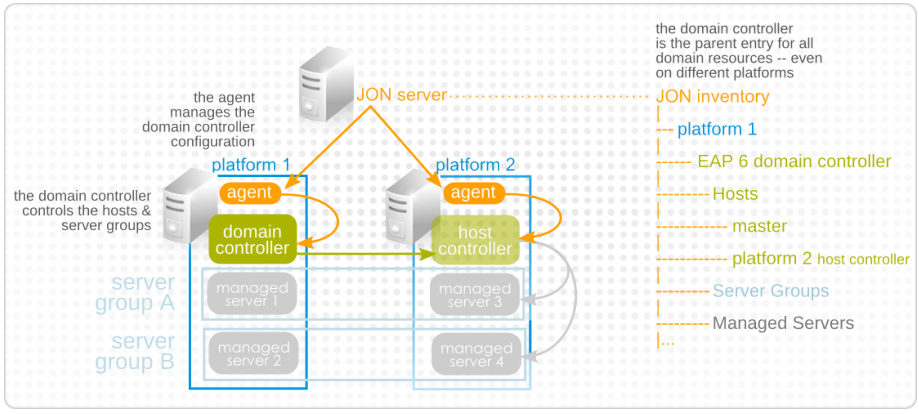


Figure 4.5. Domain Components in the JBoss ON Inventory

4.3. EAP 6 Resources in JBoss ON

Every EAP 6 configuration area in **`domain.xml`**, **`host.xml`**, or **`standalone.xml`** is interpreted as a resource type. Because EAP 6 draws a distinction between configuration components and operational components, the EAP 6 resource types in JBoss ON do different things; some resources define configuration, while others are used for monitoring, alerting, and operations.

In domains, profiles, socket bindings, and network interfaces for the domain are all configuration entries. Therefore, in JBoss ON, the corresponding resource types define configuration settings.

In domains, the managed servers are the functional instances. These resources are active, so, in JBoss ON, the managed server resources can run operations, monitor metrics, and trigger alerts.

There is no overlap between the configuration resources and the operational resources in functionality. For example, a datasource resource under a profile has configuration, but no monitoring or alerting. A datasource under a managed server collects the monitoring metrics, defines alerts, and runs operations. These resources are related — the managed server datasource resource references the configuration in the profile datasource — but they are different.

In standalone servers, configuration and management are enabled in the same resource. A standalone server is basically a combination of a profile resource (configuration) and a managed server resource (management). A datasource resource for a standalone server, then, has both configuration and monitoring, alerts, and operations on the same resource.

4.4. The Purpose of Managing EAP 6 Resources with JBoss ON

The JBoss ON UI maintains parity with the EAP 6 console, so it can be used to edit most configuration settings, create child entries, and deploy content, same as the EAP 6 management console.

The purpose of JBoss ON management, though, is not to have yet another web UI to use. JBoss ON maintains a larger perspective on the resources within the domain. This infrastructure-wide perspective reveals the relationship of a resource within the domain, the larger IT structure, and even itself and its own history aside from simply making a current configuration change.

Standalone servers and domains in JBoss ON are organized in a way that highlights the information that is available to administrators:

- Additional monitoring metrics for resources, with stored historical data, resource-specific operational baselines, and alternate display types
- Configuration and connection setting histories
- Inventory change histories when children are added or removed through JBoss ON
- Package versioning and repositories for controlled package updates
- Configuration, monitoring, and alerting for underlying and associated resources, like the platform, Apache and Tomcat web servers, and plug-ins like **`mod_cluster`**

4.5. Topics Covered in This Guide

This guide does not cover every single management options or task possible through JBoss ON. Rather, it covers tasks or configuration that are unique to EAP 6 or that require special consideration when used to manage EAP 6.

For basic management tasks, see the general JBoss ON documentation:

- "Initial Setup for the Resource Inventory, Groups, and Users" for creating dynagroups definitions, importing child resources, creating groups, and managing users and access control.
- "Deploying Applications and Content," particularly for bundles deployment.
- "Managing Resource Configuration" for managing drift and editing resource configuration.

- "Setting up Monitoring, Alerts, and Operations" for information about configuring alerts, particularly for launching script and resource operations; for setting up and changing monitoring configuration; and for running operations.
- "Writing JBoss ON Command-Line Scripts" for examples on writing server scripts and using the CLI.

Chapter 5. Upgrading the JBoss EAP 6 Resource Plug-in

JBoss Operations Network 3.0 and 3.0.1 had a tech preview version of the JBoss Enterprise Application Platform (EAP) 6 agent plug-in. With normal upgrade processes, agent plug-ins are updated when the JBoss ON server is updated. **However, tech preview plug-ins are not upgraded. The tech preview version of the plug-in must be deleted, and then the new plug-in installed.**

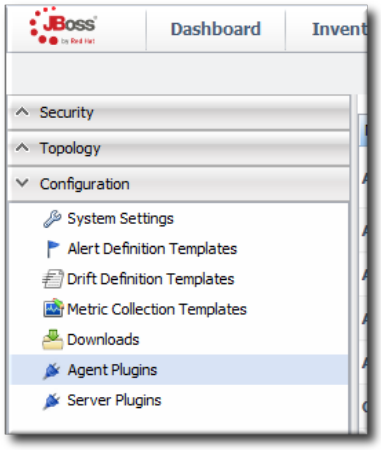
The EAP 6 tech preview plug-in and the EAP 6 GA plug-in are treated as two separate plug-ins by JBoss ON. If the tech preview version is not deleted, then any EAP 6 resource will be discovered and listed twice in the inventory, once discovered by the tech preview plug-in and discovered again by the GA plug-in.

NOTE

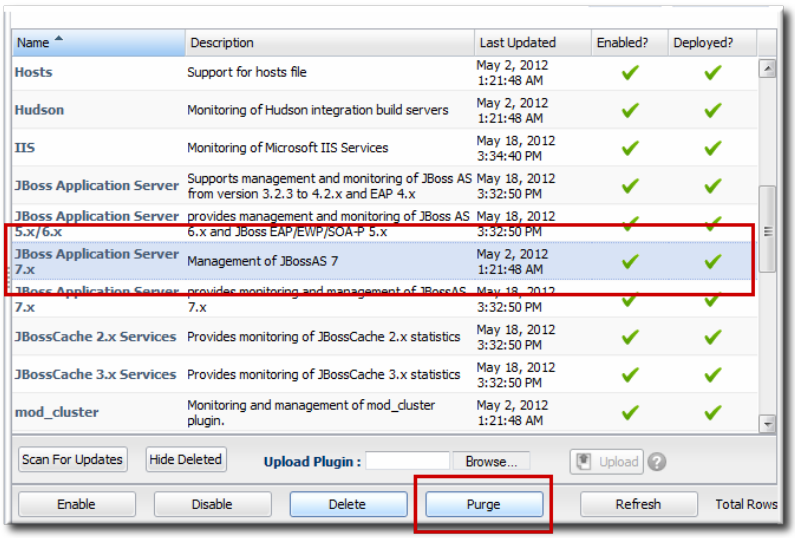
Any configuration, monitoring data and baselines, and resource histories will be lost after upgrading to the new EAP 6 plug-in.

To load the new EAP 6 plug-in:

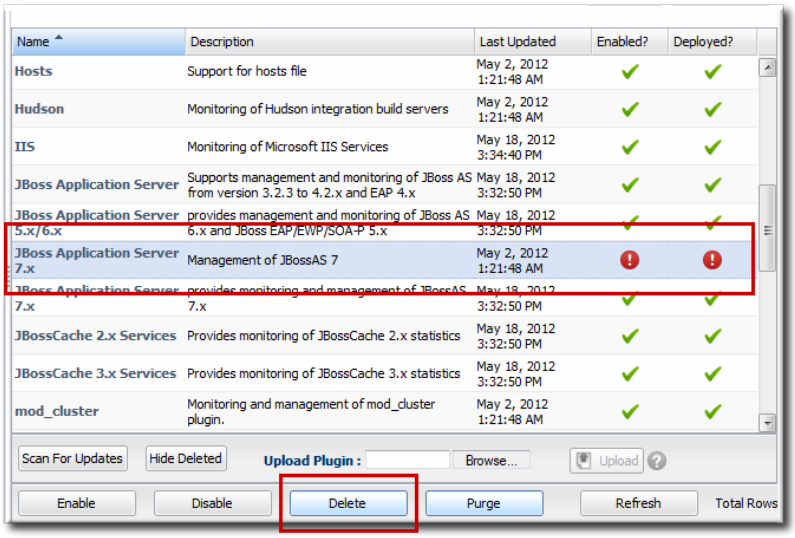
1. Delete the original tech preview plug-in and purge it from the JBoss ON database. Purging the plug-in allows the server to deploy the new plug-in in its place.
    a. In the top menu, click the **`Administration`** tab.
    b. In the **`Configuration`** box on the left navigation bar, click the **`Agent Plugins`** link.



    c. Select the EAP 6 tech preview plug-in.
    d. Click the **`Delete`** button.



    e. Click the **`SHOW DELETED`** button at the bottom of the plug-ins list.
    f. Select the EAP 6 plug-in, and then click the **`PURGE`** button. This removes the entry in the JBoss ON database that tells the servers to ignore that original tech preview plug-in and any updates to it.



IMPORTANT

Wait for the purge operation to complete before continuing with the upgrade process.

2. If the server has not already been upgraded, upgrade the JBoss ON server, as described in the server upgrade section of the Installation Guide.
3. Install the EAP 6 plug-in pack, as described in the installing plug-in packs section of the Installation Guide.
4. Import the EAP 6 server and its children, and manage the resources as normal.

Chapter 6. Setting up JBoss EAP 6 Instances

6.1. Configuration for Servers and Profiles

6.1.1. Differences for Standalone Servers and Domains

Chapter 4, *The Structure of JBoss EAP 6* goes over some of the differences between standalone servers and domain structures. The crucial difference for configuration is that **with a standalone server, all of the configuration is performed directly on the child entries.** With a domain, configuration is divided, with almost all configuration centralized in the domain-managed profiles and server groups.

This is reflected in the EAP 6 management console. Almost all configuration is under the **`Profiles`** area. The domain profiles define individual subsystem configuration, system properties, global JVM settings, and server group configuration.

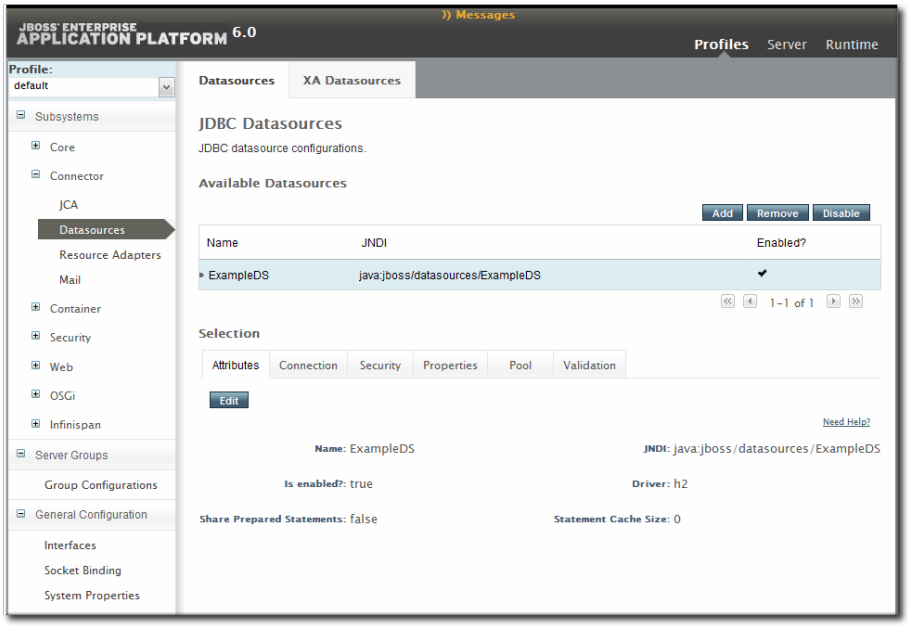


Figure 6.1. Profiles Area in the EAP 6 Console

The **`Servers`** area covers the limited amount of configuration that is set on a managed server, mainly a local JVM definition and operations like stopping and starting the server.

In JBoss ON, the major configuration areas for the domain — profiles and their subsystems, socket bindings, server groups — are broken out as separate resource types. This configuration is applied to the managed server (through the server group), much like a template.

Always edit the resource which originates the configuration settings. For most settings, that means editing the related child resource of the domain controller.

- Subsystem configuration is located in the profile resources within the **`Profiles`** autogroup for the domain controller.
- JVM definitions are configured under the domain controller (domain-wide defaults), server group (group-wide settings), or the managed server (local settings).
- Network interfaces are configured under the domain controller.
- Socket bindings themselves are configured as part of the domain controller configuration, in the entries under the **`SocketBindings`** autogroup for the domain controller. Each server group and managed server has an offset, a number that is added to the socket bindings value, which is used to give the managed servers unique port numbers in the domain; these offsets are set on the server group and managed server connection settings.
- System properties can be set on almost any server resource: the domain controller, host controller, server group, managed server.

Some configuration settings (JVM definitions and system properties) can be defined at different levels: domain, server group, or managed server. In that case, the configuration works in a cascade, with the lowest-level configuration taking precedence. So, server group configuration trumps domain settings, and managed server settings supersede server group settings. For those configuration settings, **be sure to set the configuration at the appropriate level in the domain hierarchy**. To apply settings to the entire domain, edit the relevant domain entry; to set it at the server group or server level, create or edit the configuration at that entry level.

6.1.2. Requried Management Interfaces on EAP 6

The EAP 6 plug-in in JBoss ON connects to the default HTTP management interface for the EAP 6 domain controller. This management interface is used to manage and monitor the EAP 6 domain instance.

If the HTTP management interface has been removed or disabled, the agent (using the EAP 6 plug-in) will not be able to connect to the EAP 6 domain instance. Therefore, it cannot manage or monitor the EAP 6 domain resource and the resource will appear to be unavbailable, even if it is running.

If necessary, enable the HTTP management interface for the JBoss ON agent to connect to, using the EAP 6 CLI:

```
/host=instanceName/core-service=management/management-interface=http-
interface:add(interface=http,port="\${jboss.management.http.port:9990}",security-
realm=ManagementRealm
```

6.1.3. Configuration Features in JBoss ON

JBoss ON tracks all configuration changes that are made to JBoss ON resources (through the JBoss ON UI or CLI). The emphasis on JBoss ON is not only on making configuration changes; it is on managing configuration. As with other management areas in JBoss ON, the configuration maintains its history. This allows administrators to manage configuration in the context of changes and their performance or maintenance implications:

- View the change history, including diffs between versions
- Rollback changes to any previous version, simply by clicking a button
- Track which users made changes, as part of an audit trail
- Use alerting to notify administrators of any configuration changes
- Define drift monitoring to track configuration changes against a defined baseline and to control unexpected configuration changes

For each resource, JBoss ON breaks out two types of configuration: general resource properties (in the **`Configuration`** tab) and connection properties that the agent uses to connect to the resource (in the **`Inventory`** tab). Both types of configuration have configuration histories, can be reverted to previous versions, and can be used for alerting and monitoring. In reality, editing either type of configuration could end up editing the same configuration file on the resource. These two configuration areas are separated in JBoss ON to help differentiate between the configuration that affects resource behavior and the configuration that affects connections to the resource.

Even in cases when the configuration or connection settings cannot be edited, JBoss ON will still let administrators view what configuration is being applied to that resource. This is particularly useful for managed servers, which use configuration defined in the profile resources.

6.2. Creating Management Users

The JBoss ON agents connect to the EAP 6 server as a management user. This enables JBoss ON to perform tasks like changing the configuration or starting and stopping resources.

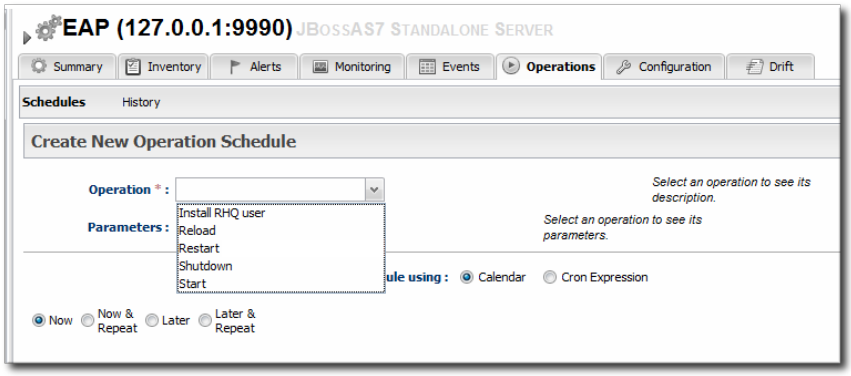
6.2.1. Creating a Management User Through JBoss ON

The operation to install a user creates the **`rhqadmin`** user in the **`ManagementRealm`** for the EAP 6 server.

NOTE

The user is created in the *secure* management realm. If the EAP 6 server is running in **un**secure mode, then the *install RHQ user* operation fails because JBoss ON cannot connect to the management realm.
EAP 6 servers are secured by default.

1. Click the **`Inventory`** tab in the top menu.
2. Select **`Servers - Top Level Imports`** in the **`Resources`** menu table on the left. Select the JBoss EAP 6 server, either the standalone server or the domain controller.
3. In the inventory tree, select the top resource entry for the server.
4. Open the **`Operations`** tab.
5. Click the **`New`** button at the bottom of the page.
6. Select the **`Install RHQ User`** option from the drop-down menu.



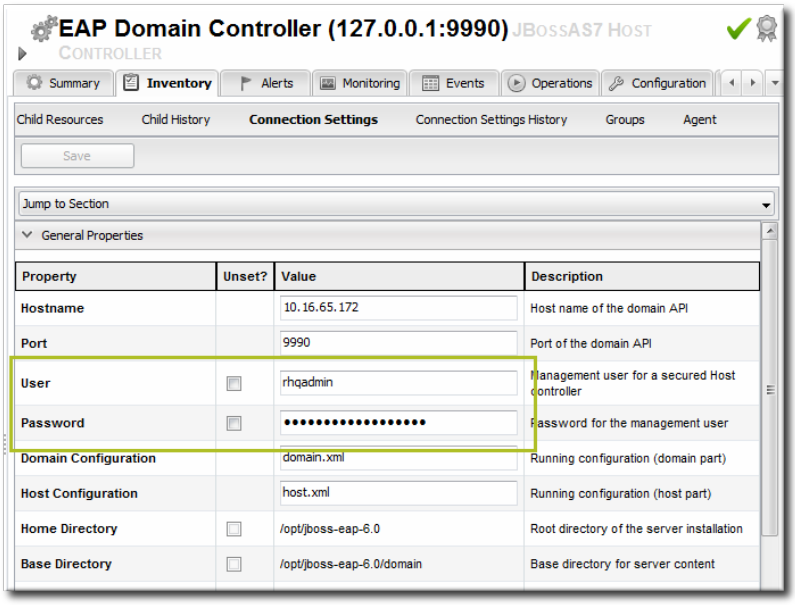
7. Click the **`Schedule`** button.

6.2.2. Creating a Management User in EAP 6

1. Run the **`add-user`** utility to create the user.

```
[root@server ~]# cd /opt/jboss-eap-6.0

[root@server bin]# ./add-user.sh
What type of user do you wish to add?
 a) Management User (mgmt-users.properties)
 b) Application User (application-users.properties)
(a): a

Enter the details of the new user to add.
Realm (ManagementRealm) :
Username : jonadmin
Password :
Re-enter Password :
About to add user 'jonadmin' for realm 'ManagementRealm'
Is this correct yes/no? yes
```

2. Set that user in the connection settings for the EAP 6 server resource in JBoss ON.
    a. Click the **`Inventory`** tab in the top menu.
    b. Select **`Servers - Top Level Imports`** in the **`Resources`** menu table on the left. Select the JBoss EAP 6 server, either the standalone server or the domain controller.
    c. In the inventory tree, select the top resource entry for the server.
    d. Open the **`Inventory`** tab.
    e. Select the **`Connection Settings`** subtab.
    f. Fill in the username and password for the management user that was created in EAP 6.



    g. Click the **`Save`** button at the top of the page.

6.3. Creating a Dynamic Group for EAP 6 Resources

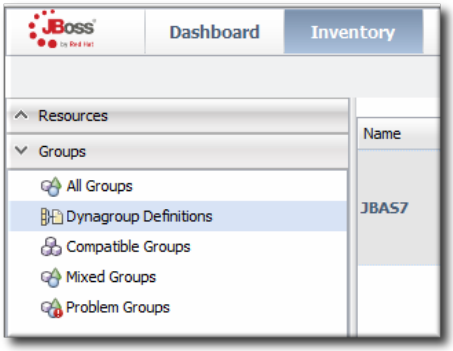
For management, particularly for monitoring or for editing configuration, it can be very convenient to have related resources grouped together in a compatible group. Compatible groups let administrators set alert definitions, change metrics settings, and change configuration for all group members simultaneously.

Dynagroups use search expressions to search for group members and then create groups based on any user-defined criteria. This assures that group membership is always current, since members are added and dropped automatically as the JBoss ON inventory changes.

The dynagroup syntax is described in much more detail in Initial Setup for the Resource Inventory, Groups, and Users. This specific dynagroup expression first searches for resources based on the resource plug-in, category (server), and parent. It then creates groups based on the product type and name.

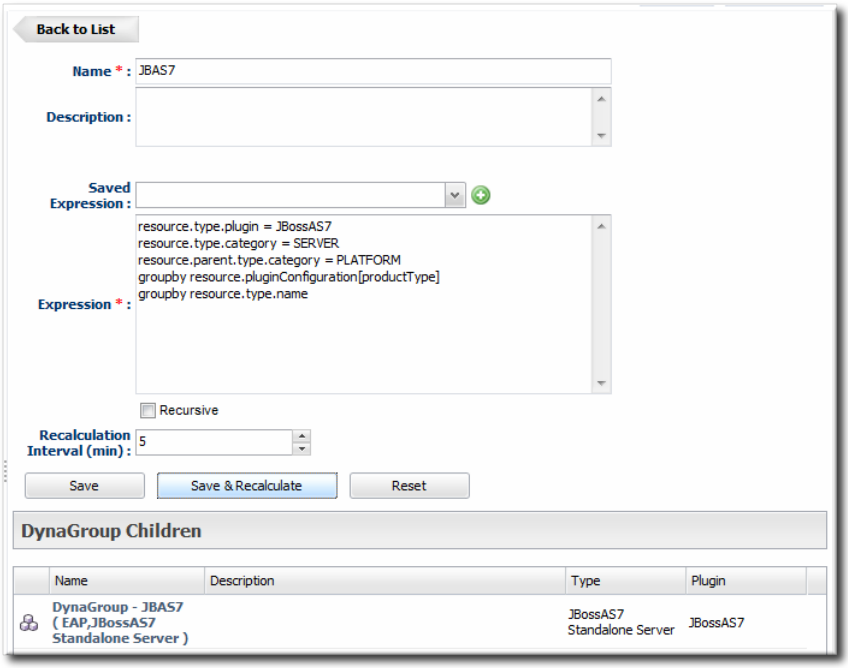
This creates a separate compatible group for every JBoss product associated with JBoss EAP 6, such as SOA-P, Data Grid (JDG), and host controllers and standalone servers.

1. Click the **`Inventory`** tab in the top menu.
2. In the **`Groups`** area on the left, click the **`Dynagroup Definitions`** link.



3. Enter the expression to create compatible groups for each EAP 6 server type.

```
resource.type.plugin = JBossAS7
resource.type.category = SERVER
resource.parent.type.category = PLATFORM
groupby resource.pluginConfiguration[productType]
groupby resource.type.name
```



4. Click the **`Save`** button in the middle of the page.

6.4. Setting Start Script Arguments, Environment Variables, and JAVA_OPTS

6.4.1. Start Script Discovery and Settings

As part of discovering a JBoss EAP 6 server (domain controller or standalone server), JBoss ON attempts to discover the environment that the server is running in. Specifically, the discovery process attempts to identify and recreate the runtime environment:

- The discovery process identifies, or attempts to identify, the start script used, including custom start scripts.

- Discovery detects a subset of environment variables set in the **run.conf** file or parent process that are required for the start script to work.

> **NOTE**
>
> Although the discovery process does detect some environment variables, **the discovery scan does not detect JAVA_OPTS values.**
> The connection properties for the start script intentionally defer to the **run.conf** file for **JAVA_OPTS** values.

- Discovery attempts to detect any arguments passed with the start script itself.
- Discovery attempts to detect what user the script is running as and assign a *prefix* command to use with the start script. For example, if the start script is running as the **jboss** user and the JBoss ON agent is running as **jonagent**, then the discovery script automatically assigns a **sudo** command, **sudo -u jboss -g jboss**, to pass with the start script.

The discovered settings are stored in the **Start Script Environment Variables** and **Start Script Arguments** connection settings for the EAP 6 resource.

If the discovery scan cannot detect some of the script settings (such as the agent is running as a different user than the EAP 6 server and cannot read the parent process), it fails gracefully. It simply gathers whatever information it can and ignores blank values.

> **NOTE**
>
> The **Start Script Environment Variables** and **Start Script Arguments** connection settings are only discovered once, when the resource is initially discovered. After that, the connection settings can be changed in the connection settings, but any changes made on the local system will not be detected. For example, if the server is discovered with a particular value (like **-XX:PermSize=256M**), the argument value will not be updated if the server is restarted later with a different setting value.

The script arguments and environment variables are the only ones used by the agent when it runs the start script. These arguments and environment variables are not added to other configuration settings or the parent process. The start script settings in the EAP 6 connection settings configuration are deterministic.

6.4.2. Start Script Arguments and Drift Monitoring

JBoss ON can monitor directories or specific files to check for *configuration drift*, which means configuration changes that move away from a designated configuration state. This is described in more detail in Section 6.12, “Controlling Configuration Drift” and the drift chapters of "Managing Resource Configuration."

While drift monitoring is critical for administrators to manage important resources, there may be times when it is necessary or beneficial from that configuration in a few settings. The start script arguments can be used to override the defined configuration for an EAP 6 server *without* triggering a drift alert.

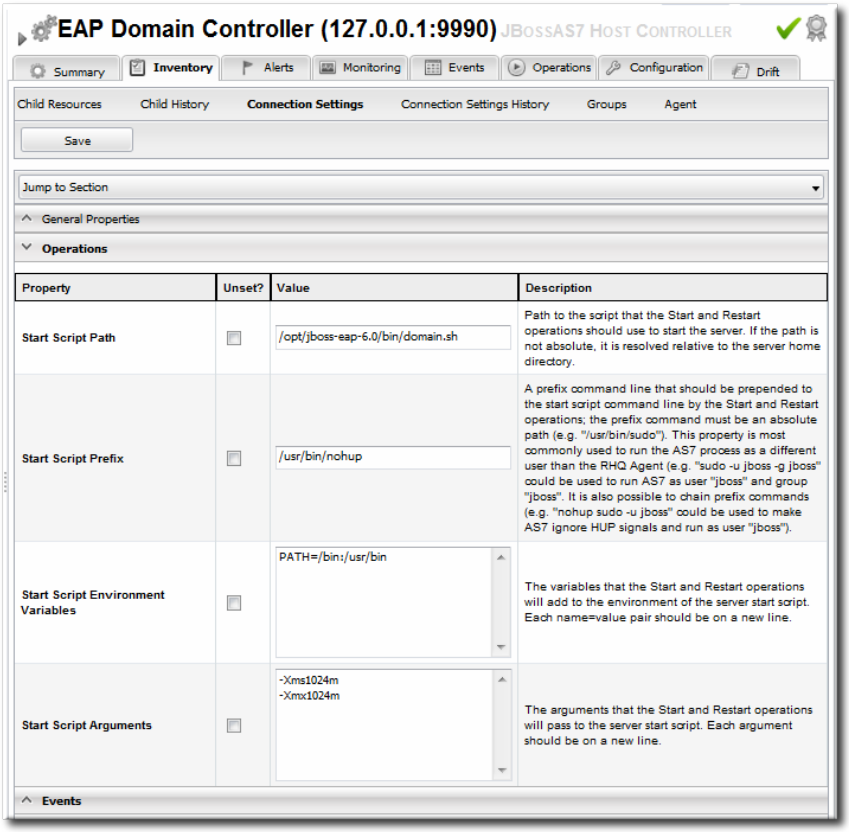
Since the start script options are all connection settings, every change is recorded in a change history and can be easily viewed and reverted, as in Section 6.13, “Tracking and Reverting Configuration Changes”. This keeps the system properties and Java settings trackable and remediable.

> **Example 6.1. System Properties Without Violating the Drift Definition**
>
> Tim the IT Guy creates one specific set of configuration that all production EAP 6 servers should use an environment. He then creates a drift definition template for monitoring and associates or pins that blessed configuration to the template.
>
> Every EAP 6 server which uses that drift template must conform to those configuration settings. Tim the IT Guy creates an alert that informs him if any production server drifts away from that specified configuration. This is an important safety measure for Example Co.'s production application servers.
>
> However, a couple of the production servers have slightly different hardware and other applications running on them, so they require different heap sizes to run effectively.
>
> If Tim the TI Guy adds a system property to use a different heap size, he is going to receive constant drift alerts or his edited configuration could be overwritten if he runs an automatic server-side script to remediate drift configuration.
>
> By setting different heap settings through the start script, Tim can apply the right settings for that system without editing a configuration file, so there is no alert-able configuration drift.

6.4.3. Changing Start Script Configuration

1. Click the **Inventory** tab in the top menu.

2. Select **`Servers - Top Level Imports`** in the **`Resources`** menu table on the left. Select the JBoss EAP 6 server.
3. In the inventory tree, select the top resource entry for the server.
4. Open the **`Inventory`** tab, and select the **`Connection Settings`** subtab.
5. Expand the **`Operations`** area.
6. Change or add start script settings. These are the scripts and settings that the JBoss ON agent uses when running a start or restart operation on the EAP 6 server.


EAP Domain Controller (127.0.0.1:9990)
JBossAS7 Host Controller
Summary
Inventory
Alerts
Monitoring
Events
Operations
Configuration
Drift
Child Resources
Child History
Connection Settings
Connection Settings History
Groups
Agent
Save
Jump to Section
General Properties
Operations
Property
Unset?
Value
Description
Start Script Path
/opt/jboss-eap-6.0/bin/domain.sh
Path to the script that the Start and Restart operations should use to start the server. If the path is not absolute, it is resolved relative to the server home directory.
Start Script Prefix
/usr/bin/nohup
A prefix command line that should be prepended to the start script command line by the Start and Restart operations; the prefix command must be an absolute path (e.g. "/usr/bin/sudo"). This property is most commonly used to run the AS7 process as a different user than the RHQ Agent (e.g. "sudo -u jboss -g jboss" could be used to run AS7 as user "jboss" and group "jboss". It is also possible to chain prefix commands (e.g. "nohup sudo -u jboss" could be used to make AS7 ignore HUP signals and run as user "jboss").
Start Script Environment Variables
PATH=/bin:/usr/bin
The variables that the Start and Restart operations will add to the environment of the server start script. Each name=value pair should be on a new line.
Start Script Arguments
-Xms1024m
-Xmx1024m
The arguments that the Start and Restart operations will pass to the server start script. Each argument should be on a new line.
Events

   - To use a custom start script, one other than **`domain.sh`** or **`standalone.sh`**, enter the path and script name.
   - Optionally, enter a prefix to use with the script when running the start script.

     When the start script is discovered, the agent tries to determine the user the script is running as and assign a prefix command to use with the start script. For example, if the start script is running as the **`jboss`** user and the JBoss ON agent is running as **`jonagent`**, then the discovery script automatically assigns a **`sudo`** command, **`sudo -u jboss -g jboss`**, to pass with the start script.

     Additionally, JBoss ON assigns the **`nohup`** command as a prefix so that if the JBoss EAP/AS server is started by the agent and the agent process dies, the JBoss EAP/AS server process continues running.
   - Set any environment variables, one per line.
   - Set any script arguments, one per line. For regular JAVA_OPTS, these arguments usually are **`-X`**, **`-D`**, or **`-P`**. Some useful **`-XX`** arguments are listed in the JVM options documentation from Sun. Some useful system properties for EAP 6 are listed with the JBoss AS7 project documentation.

     The EAP 6 default start scripts use a **`run.sh`**-style script, so the arguments use that format. A custom script can use different arguments or options.
7. Click the **`Save`** button at the top of the page.

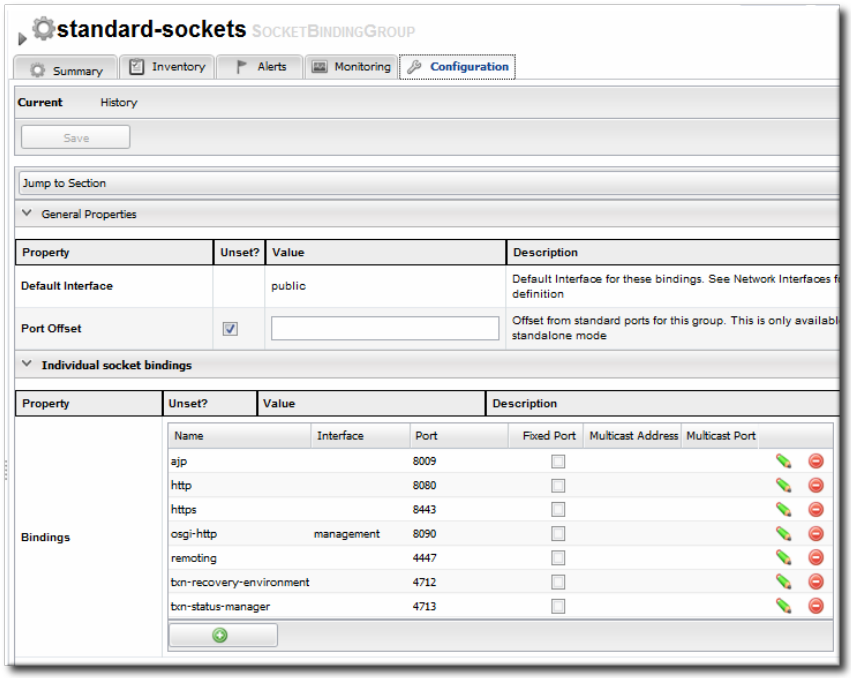
6.5. Changing Port Numbers

6.5.1. Changing Socket Binding Ports

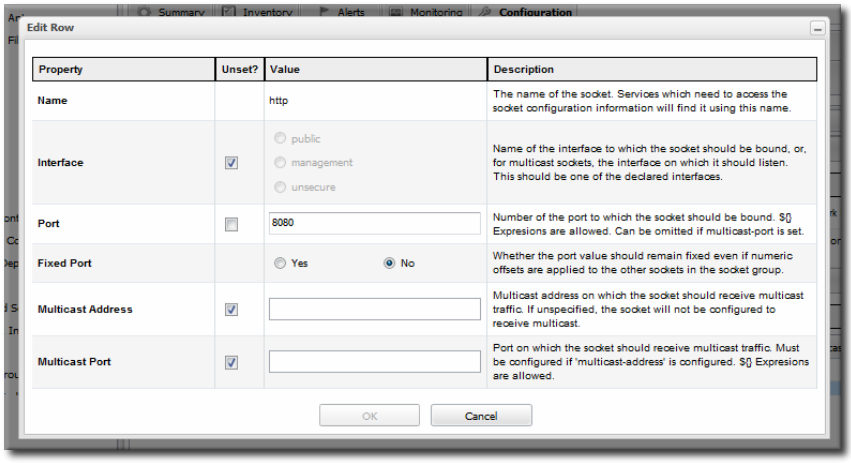
The socket binding resources defines both what ports are available (such as HTTP, AJP, and HTTPS) and what those port numbers are. The socket binding configuration can also configure multicast port numbers for the sockets.

1. Click the **`Inventory`** tab in the top menu.
2. Select **`Servers - Top Level Imports`** in the **`Resources`** menu table on the left. Select the JBoss EAP 6 server.
3. In the inventory tree, select the **`SocketBindingsGroup`** compatible group, and then select the socket binding to edit.

4. Open the **Configuration** tab.
5. Click the green pencil icon to edit an existing socket definition or click the green plus sign (+) to create a new one.



6. Change the **Port** number to any available port between 1025 and 65535. On Linux, available port numbers can be determined using **iptables**.


Edit Row
Property
Unset?
Value
Description
Name
http
The name of the socket. Services which need to access the socket configuration information will find it using this name.
Interface
public
management
unsecure
Name of the interface to which the socket should be bound, or, for multicast sockets, the interface on which it should listen. This should be one of the declared interfaces.
Port
8080
Number of the port to which the socket should be bound. ${} Expresions are allowed. Can be omitted if multicast-port is set.
Fixed Port
Yes
No
Whether the port value should remain fixed even if numeric offsets are applied to the other sockets in the socket group.
Multicast Address
Multicast address on which the socket should receive multicast traffic. If unspecified, the socket will not be configured to receive multicast.
Multicast Port
Port on which the socket should receive multicast traffic. Must be configured if 'multicast-address' is configured. ${} Expresions are allowed.
OK
Cancel

   Optionally, configure multicast settings for the socket. If there are multiple instances of JBoss servers on the same system or in the same cluster, then multicast may be configured for cluster communication.
7. Click the **Save** button at the top of the page.

6.5.2. Changing Port Offsets for Server Groups in a Domain

The port offset is a number added to each port number in a socket binding group to create the actual port numbers used by a server instance. This allows managed servers to have unique port numbers across the domain or for standalone servers within a cluster to have unique ports while using the same socket binding configuration.

For example, if the socket binding HTTP port is 8080, and a managed server has an offset of 110, the actual port used by the managed server is 8190 (8080 + 110). Server groups can also define offsets, and the offsets are additive. So, if a server group has an offset of 15 and the managed server has an offset of 110, the port number used by the server is 8205 (8080 + 15 + 110).

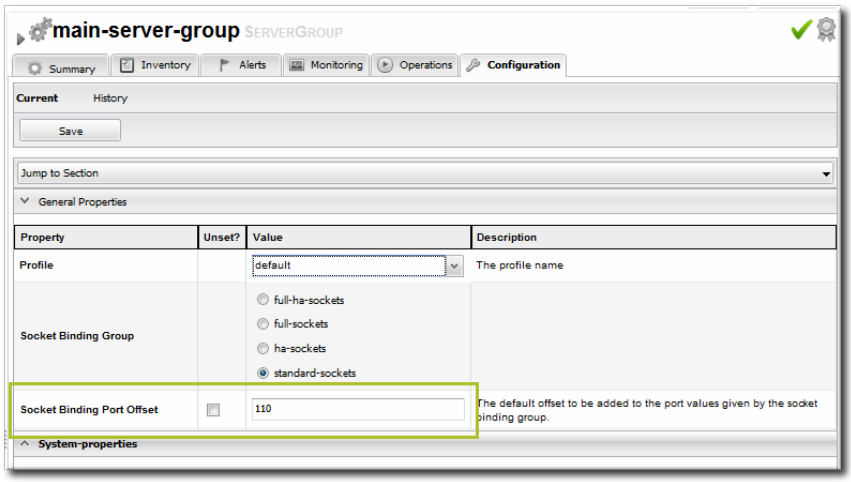
NOTE

The offset for a managed server is defined in its entry in the **host.xml** file. This can be set when the managed server is created in JBoss ON, but it cannot be edited afterward.

It is possible to edit the port offset used by a server group:

1. Click the **Inventory** tab in the top menu.

2. Select **`Servers - Top Level Imports`** in the **`Resources`** menu table on the left. Select the JBoss EAP 6 server.
3. In the inventory tree, expand the **`Server Groups`** node, and select the server group.
4. Open the **`Configuration`** tab for the server group.
5. In the **`Port Offset`** field, enter the new value for the offset.



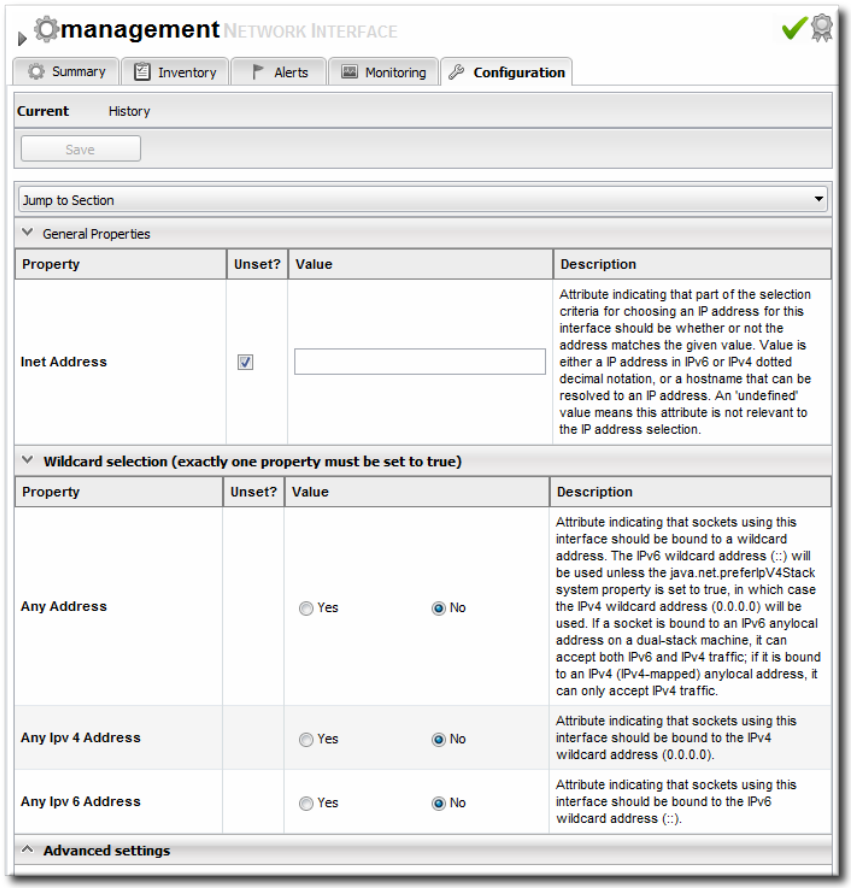
6. Click **`Save`** at the top of the page.

To change the offset for a standalone server, edit its connection settings, as in Section 6.9.1, “Changing the General Properties for an EAP 6 Server”.

6.6. Editing Network Interfaces

Network interfaces can be configured to use a specific IP address or a type of IP address when communicating with sockets for that interface.

1. Click the **`Inventory`** tab in the top menu.
2. Select **`Servers - Top Level Imports`** in the **`Resources`** menu table on the left. Select the JBoss EAP 6 server.
3. In the inventory tree, select the **`Network Interfaces`** group under the main server entry, and select the interface (management, public, or unsecure).
4. Open the **`Configuration`** tab.
5. Set either the specific IP address for the interface to use *or* set which type of IP address to use (IPv4, IPv6, or either). **Either the IP address or the IP address type must be set.**


management NETWORK INTERFACE
Summary
Inventory
Alerts
Monitoring
Configuration
Current
History
Save
Jump to Section
General Properties
Property
Unset?
Value
Description
Inet Address
Attribute indicating that part of the selection criteria for choosing an IP address for this interface should be whether or not the address matches the given value. Value is either a IP address in IPv6 or IPv4 dotted decimal notation, or a hostname that can be resolved to an IP address. An 'undefined' value means this attribute is not relevant to the IP address selection.
Wildcard selection (exactly one property must be set to true)
Property
Unset?
Value
Description
Any Address
Yes
No
Attribute indicating that sockets using this interface should be bound to a wildcard address. The IPv6 wildcard address (::) will be used unless the java.net.preferIpV4Stack system property is set to true, in which case the IPv4 wildcard address (0.0.0.0) will be used. If a socket is bound to an IPv6 anylocal address on a dual-stack machine, it can accept both IPv6 and IPv4 traffic; if it is bound to an IPv4 (IPv4-mapped) anylocal address, it can only accept IPv4 traffic.
Any Ipv 4 Address
Yes
No
Attribute indicating that sockets using this interface should be bound to the IPv4 wildcard address (0.0.0.0).
Any Ipv 6 Address
Yes
No
Attribute indicating that sockets using this interface should be bound to the IPv6 wildcard address (::).
Advanced settings

Because either a specific IP address or an IP address type can be set, and which property is used is optional, the UI does not enforce that a selection is made. For the network interface to work properly, however, some kind of IP address configuration must be set.

6. Click the **Save** button at the top of the page.

6.7. Setting System Properties

Servers and particularly subsystems, which are integrated services, can have additional configuration properties added as system properties. The system property is a simple attribute-value pair, and it can be anything. The system properties which are available depend on the resource being edited.

When editing profiles and subsystems, extensions, server groups, and other domain components, system properties are added to the component's entry in the **`domain.xml`** file. When editing a host controller or a managed server, the properties are added to the server's entry in the **`host.xml`** file.

TIP

System properties, as with other configuration elements in the domain, work in a cascade. If a system property is set on a server group, it applies to all members of the group. If it is set on a managed server, it applies only to that server.
Be sure to edit the entry at the appropriate level in the domain so that the configuration is applied appropriately.

System properties set default configuration; these can be overridden with the start script in the **`-D`** or **`-P`** arguments.

NOTE

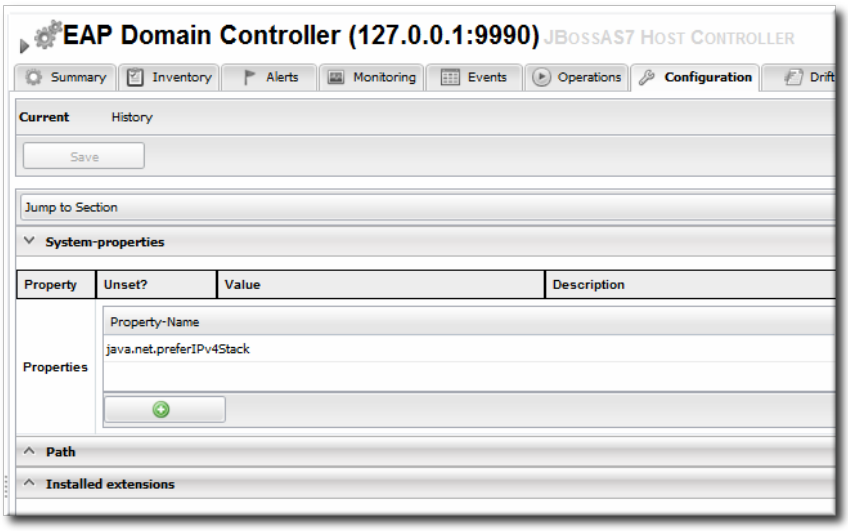
If drift monitoring is configured, then adding or editing system properties could trigger a drift alert. See Section 6.12, “Controlling Configuration Drift”.

Check the project documentation for information on available system properties:

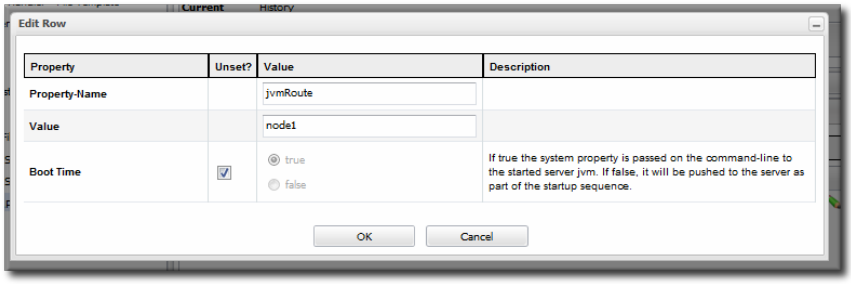
- JBoss AS7
- mod_cluster
- JBossWeb

To add system properties to the configuration:

1. Click the **Inventory** tab in the top menu.
2. Select **Servers - Top Level Imports** in the **Resources** menu table on the left. Select the JBoss EAP 6 server.
3. In the inventory tree, select the top resource entry for the server.
4. Open the **Configuration** tab.
5. Expand the **Properties** section.



6. Click the green plus (+) icon at the bottom of the **Paths** list.
7. Fill in the new property information.



   - The system property name.
   - The value of the property.
   - If the property should be loaded immediately to the running JVM or if it should be loaded when the JVM is started. The default is to load it immediately.
8. Click **OK**.

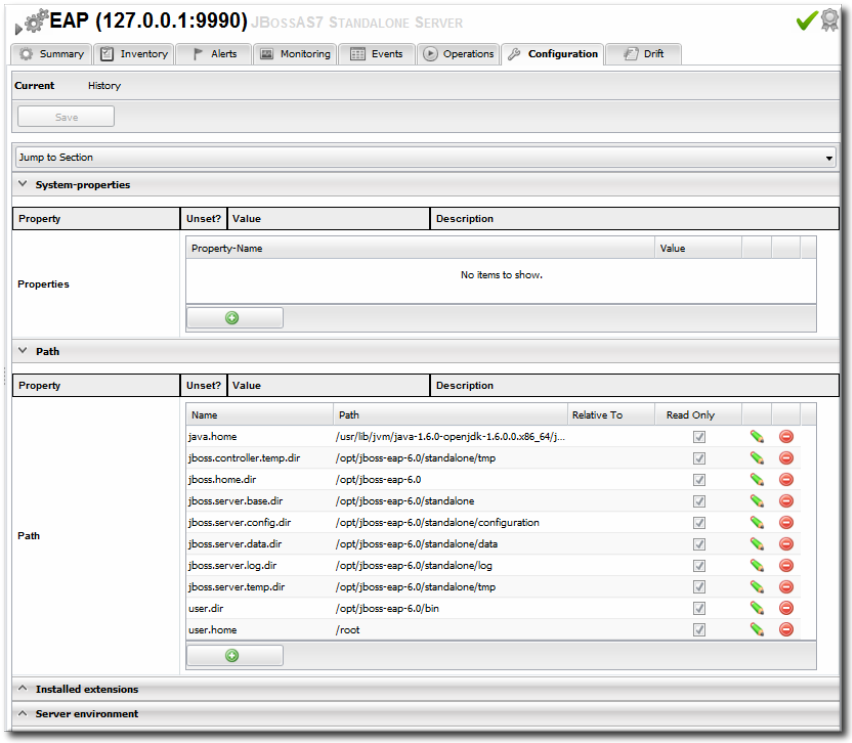
6.8. Adding System Paths

Some system paths are already defined for important server directory locations, like the home directory, log directory, and Java home directory. For custom applications, it may be useful or necessary to define other paths, and these can be added to the server configuration.

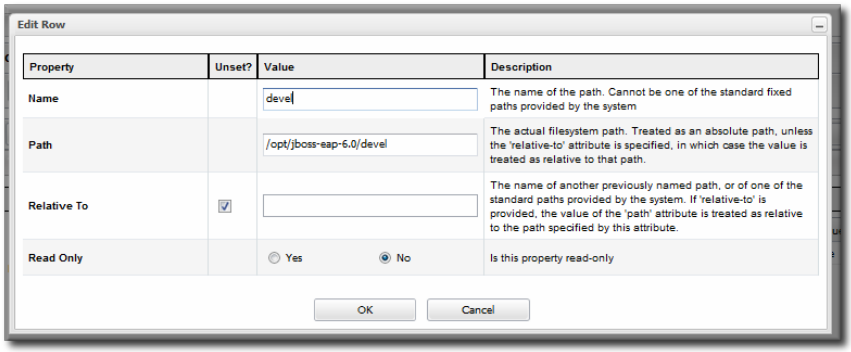
NOTE

The default system paths cannot be edited or deleted through the resource configuration. These are defined by the JBoss EAP 6 installation itself. These paths begin with the names **jboss.***, **user.***, and **java.***.
Some of those default system paths, like the home directory and base directory, can be edited by editing the EAP 6 server connections settings, as in Section 6.9, “Editing Connection Settings”.
Edit and delete icons are displayed by these default path entries. Although an edit window comes up and the path can apparently be edited or deleted, those changes are reset immediately.

1. Click the **Inventory** tab in the top menu.
2. Select **Servers - Top Level Imports** in the **Resources** menu table on the left. Select the JBoss EAP 6 server.
3. In the inventory tree, select the top resource entry for the server.
4. Open the **Configuration** tab.
5. Expand the **Paths** section.



6. Click the green plus (+) icon at the bottom of the **Paths** list.
7. Fill in the path information.


Edit Row
Property
Unset?
Value
Description
Name
devel
The name of the path. Cannot be one of the standard fixed paths provided by the system
Path
/opt/jboss-eap-6.0/devel
The actual filesystem path. Treated as an absolute path, unless the 'relative-to' attribute is specified, in which case the value is treated as relative to that path.
Relative To
The name of another previously named path, or of one of the standard paths provided by the system. If 'relative-to' is provided, the value of the 'path' attribute is treated as relative to the path specified by this attribute.
Read Only
Yes
No
Is this property read-only
OK
Cancel

   - The name of the path to create.
   - The path (absolute or relative) to create.
   - If a relative path was given as the **Path** value, then de-select the **Unset?** checkbox for the **Relative** field, and enter the *name* of the system path that it is relative to.

     For example, if the new path is **devel/**, and this is relative to the EAP home directory, then the **Relative** value is *java.home.dir*. This results in a final path of **/opt/jboss-eap-6.0/devel/**.
   - If the property is read-only. A read-only property **cannot** be edited after it is created. Read-only paths (aside from the default paths) have to be deleted and recreated if they need to be changed.
8. Click **OK**.

6.9. Editing Connection Settings

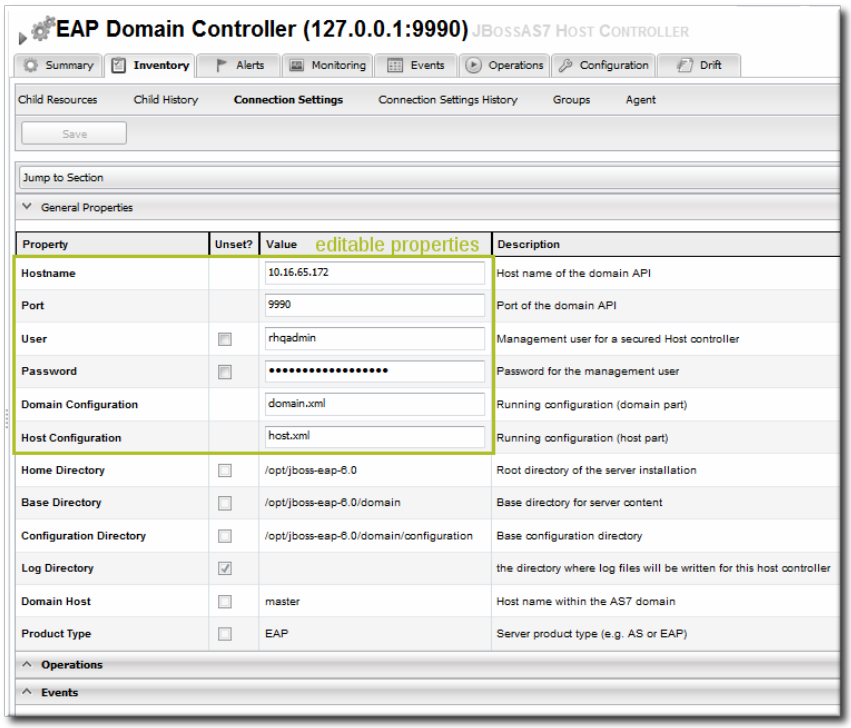
6.9.1. Changing the General Properties for an EAP 6 Server

The connection settings are the properties that the JBoss ON agent uses to connect to a resource; the connection settings to use are identified in the plug-in descriptor for the resource.

Connection settings are only configurable for a domain controller or standalone server; all of the child resource connection properties are derived from the main server configuration.

1. Click the **Inventory** tab in the top menu.
2. Select **Servers - Top Level Imports** in the **Resources** menu table on the left. Select the JBoss EAP 6 server.
3. In the inventory tree, select the top resource entry for the server.
4. Open the **Inventory** tab, and select the **Connection Settings** subtab.
5. The server connection properties are in the **General Properties** section. Only some of the

properties can be edited. Information that is derived from the JBoss EAP 6 installation itself, like the home directory, base directory, and server type (EAP or AS) is displayed, but is inactive.



- **`Hostname`** gives IP address to use to connect to the server. This is usually 127.0.0.1, but if the management interface configuration has been changed, then the IP address may be a public IP instead of the localhost.
- **`Port`** is the port of the management interface.
- **`Username`** and **`Password`** are the credentials of the JBoss EAP 6 user for the agent to use to log in. If this user was created using the *install RHQ user* operation, then the user is **`rhqadmin`**.
- *Domain Controllers Only.* With the standalone server, all of the configuration and the server instance definition are in the same file, **`standalone.xml`** or any other configuration file passed to the start script. For domains, the server configuration is defined in one file (for the domain controller), while the server instances are defined in a separate file (for the host controller). The **`Domain Configuration`** and **`Host Configuration`** fields give the names of the files within the **`domain/configuration/`** directory to reference for profile configuration and for managed server instances, respectively.

6. Click the **`Save`** button at the top of the page.

NOTE

Along with displaying the information in the **`Connection Settings`** area, the settings for a standalone server are also displayed (but not editable) in the **`Configuration`** tab, under the **`Server Environment`** area.

6.9.2. Viewing Installation Paths for EAP 6 Child Resources

Child resources for EAP 6 server such as managed servers, subsystem services, JVM definitions, and server groups are defined within the EAP 6 server's configuration files.

The "connection setting" for the child resource, then, is the entry in that server's configuration.

The connection setting is viewable in the child resource's **`Inventory > Connection Settings`** tab, but it cannot be edited because it is intrinsic to the resource itself. The connection setting for the child is the resource.

For example, the **`main-server-group`** definition is in the domain controller's **`domain.xml`** file:

```
    <server-groups>
        <server-group name="main-server-group" profile="full">
... 8< ...
```

In the JBoss ON GUI, the definition is given as the connection setting path, in the form *element=name*

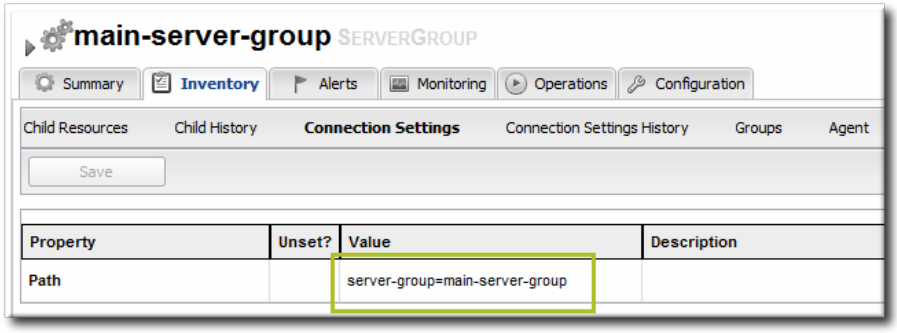


Figure 6.2. Child Resource Connection Settings

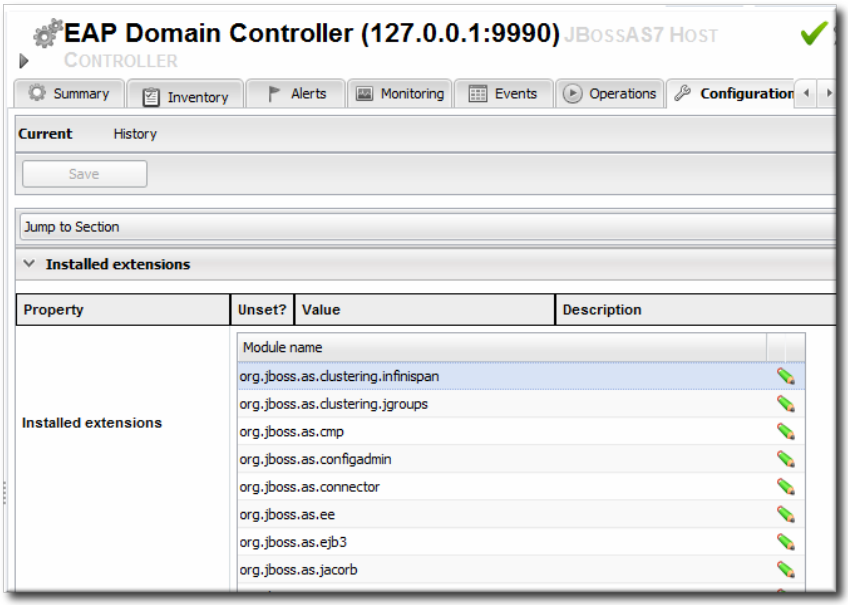
6.10. Viewing Installed Extensions

All of the subsystems available to JBoss EAP 6 profiles, both for standalone servers and for domains, are loaded as *extensions*. These extensions are modules, classes that are defined in the configuration file (**`domain.xml`** or **`standalone.xml`**).

```
    <extensions>
        <extension module="org.jboss.as.clustering.infinispan"/>
        <extension module="org.jboss.as.clustering.jgroups"/>
        <extension module="org.jboss.as.cmp"/>
        <extension module="org.jboss.as.configadmin"/>
        <extension module="org.jboss.as.connector"/>
        <extension module="org.jboss.as.ee"/>
        <extension module="org.jboss.as.ejb3"/>
... 8< ...
```

These extensions are not configurable through JBoss ON, but the current extension configuration can be viewed in JBoss ON as part of the standalone server or domain controller configuration.

1. Click the **`Inventory`** tab in the top menu.
2. Select **`Servers - Top Level Imports`** in the **`Resources`** menu table on the left. Select the JBoss EAP 6 server.
3. In the inventory tree, select the top resource entry for the server.
4. Open the **`Configuration`** tab.
5. Expand the **`Installed extensions`** section.



NOTE

An edit icon appears next to module names in the **`Extensions`** area of the EAP 6 server configuration. These properties are read-only and cannot, in fact, be edited even though an edit box pops up.

6.11. Reloading the Server Configuration

Some configuration changes require that the EAP 6 server be restarted before they take effect. The server is marked internally as being in a *requires-reload* state. JBoss ON does not force a reload, but it does inform administrators if changes have been made that require a reload before they can be applied.

TIP

Because JBoss ON does not automatically reload configuration every time a configuration change is made, administrators can make multiple changes and then schedule a time for them to be applied by scheduling the reload operation during a maintenance window (through JBoss ON) or by setting up a cron job on the local system.

Clicking *any* **`Configuration > Current`** tab, for any resource within the server tree, brings up a message box that the server must be reloaded.

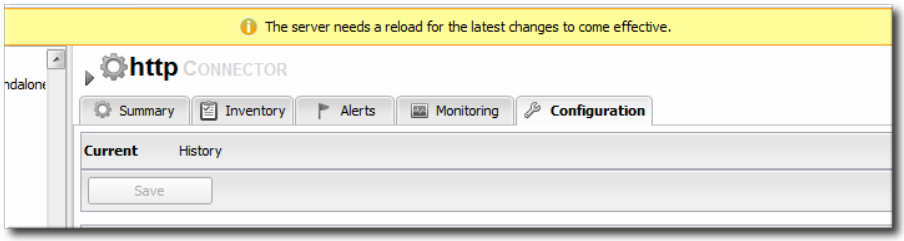


Figure 6.3. Reload Configuration Message

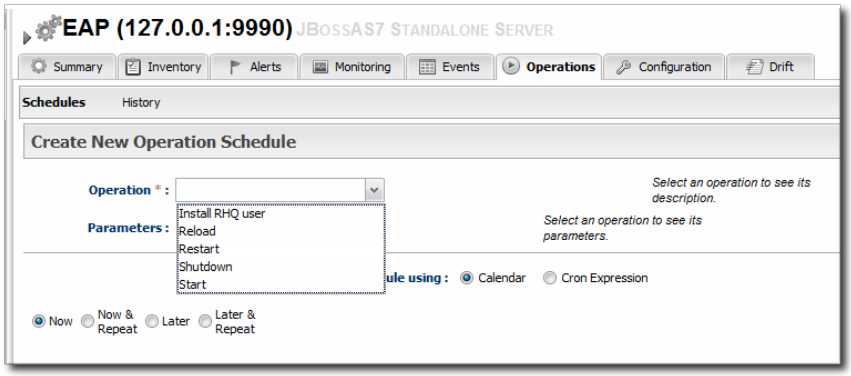
TIP

Changes that require the configuration to be reloaded typically involve changing the way that connections are handled, such as resetting port numbers or changing connection protocols for an interface.

NOTE

If the agent goes offline and a user views a **`Configuration`** tab in the UI, the message box is displayed, even if the server has been reloaded. This is because the agent was not able to communicate to the server that the reload state has changed, so the server displays outdated information.

1. Click the **`Inventory`** tab in the top menu.
2. Select **`Servers - Top Level Imports`** in the **`Resources`** menu table on the left. Select the JBoss EAP 6 server.
3. In the inventory tree, select the top resource entry for the server.
4. Open the **`Operations`** tab.
5. Click the **`New`** button at the bottom of the page.
6. Select the **`Reload`**) option from the drop-down menu.



7. Click the **`Schedule`** button.

6.12. Controlling Configuration Drift

Configuration drift is the accumulation of changes, over time, to the desired or administrator-defined configuration. These changes can be trivial, accidental, or incremental, but it moves the resource outside the intended settings.

Configuration drift and JBoss ON's drift monitoring are covered in detail in the drift chapters of "Managing Resource Configuration."

For EAP 6, plan a drift strategy that covers all of the critical configuration and provides a path to remediation, possibly without requiring administrator intervention, to help preserve production systems:

1. Set drift definitions that track the critical configuration directories, such as **`domain/configuration/`** and **`standalone/configuration/`**, but that exclude directories which will have constantly changing data, such as logging, library, and data directories. Even within the configuration directories, create exclude rules for the **`host_xml_history/`**, **`domain_xml_history/`**, and **`standalone_xml_history/`** directories, since those are not proper configuration files and should not be tracked.
2. Once the desired configuration is in place, *pin* that configuration to the drift definition. This sets the desired configuration as the baseline. All changes will be compared against that baseline.
3. Create an archive of the blessed configuration.
4. Create a bundle definition that can be automatically deployed to reset the EAP 6 configuration and remediate drift.

   When creating the he destination should be the platform of the EAP 6 resource. The destination could be the standalone server or the domain controller, but using the platform allows you to deploy the bundle to an expendable directory, like **`/tmp/mybundles/holding`**, and then run a post-install task that copies the configuration files into the configuration directory.

   Deploying a bundle generally removes whatever existing files are in the target directory and replaces them with the bundle. There are ways to control that behavior, but, generally, it is safest to have the contents of the bundle match exactly what the final deployment will be.

   Since it may not be feasible to have the entire configuration directory in the bundle, deploying to a separate location on the filesystem preserves the configuration directory, and only the important configuration files are updated (when they are copied by the Ant task).

   For more on bundles and remedying drift, see the bundles chapter in "Deploying Applications and Content" and the drift-bundle CLI example script in "Writing JBoss ON Command-Line Scripts."
5. Set up alerts for configuration drift that do two things:
   - Send a notification email to administrators.
   - Run a CLI script on the platform that automatically deploys the bundle.

   "Setting up Monitoring, Alerts, and Operations" has information on how to configure alert notifications that launch a JBoss ON server-side script or that run an operation on another resource.

NOTE

Changing the EAP 6 resource configuration, changing JVM definition settings, adding server groups and managed servers, or changing configuration settings all edit the EAP 6 configuration files, **`domain.xml`** and **`standalone.xml`**. That will trigger a drift alert, if alerting is configured.

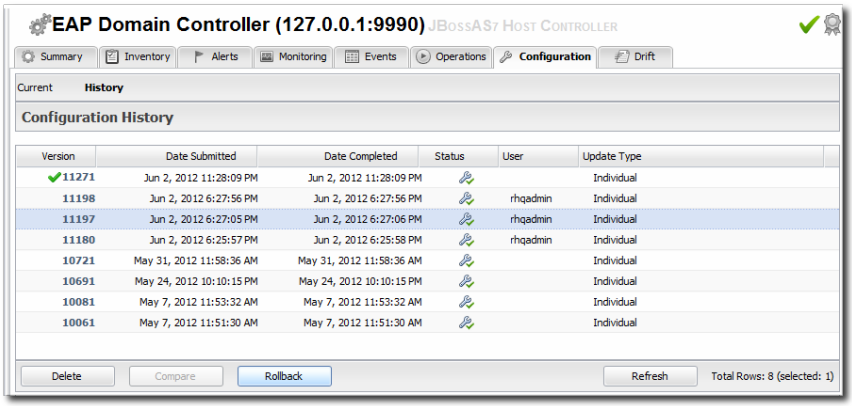
6.13. Tracking and Reverting Configuration Changes

Every configuration change made through the JBoss ON UI or CLI is recorded in a change history.

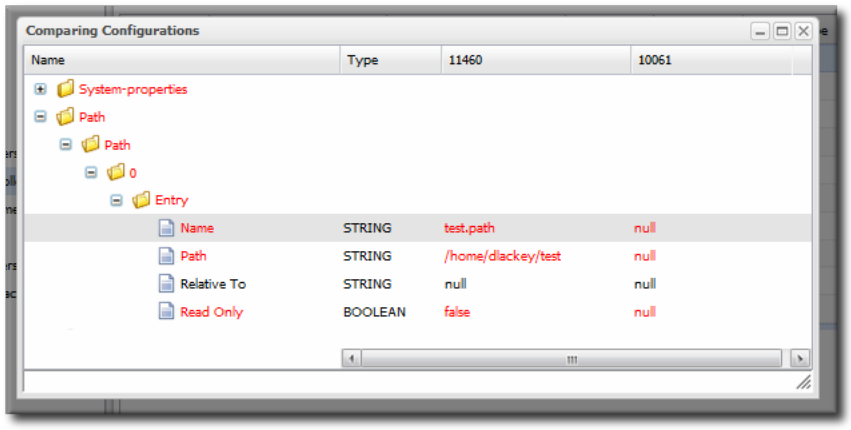
1. Click the **`Inventory`** tab in the top menu.
2. Select **`Servers - Top Level Imports`** in the **`Resources`** menu table on the left. Select the JBoss EAP 6 server.
3. In the inventory tree, select the top resource entry for the server.
4. Open the **`Configuration`** tab, and select the **`History`** subtab.

   NOTE

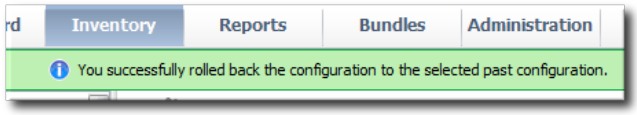
   Change history pages are kept for resource configuration (the **`Configuration`** tab) and the connection settings (the **`Inventory > Connection Settings`** tab).
5. Clicking the change ID number opens the configuration settings that were in effect *for that version*.


EAP Domain Controller (127.0.0.1:9990)
JBossAS7 Host Controller
Summary
Inventory
Alerts
Monitoring
Events
Operations
Configuration
Drift
Current
History
Configuration History
Version
Date Submitted
Date Completed
Status
User
Update Type
11271 Jun 2, 2012 11:28:09 PM Jun 2, 2012 11:28:09 PM Individual
11198 Jun 2, 2012 6:27:56 PM Jun 2, 2012 6:27:56 PM rhqadmin Individual
11197 Jun 2, 2012 6:27:05 PM Jun 2, 2012 6:27:06 PM rhqadmin Individual
11180 Jun 2, 2012 6:25:57 PM Jun 2, 2012 6:25:58 PM rhqadmin Individual
10721 May 31, 2012 11:58:36 AM May 31, 2012 11:58:36 AM Individual
10691 May 24, 2012 10:10:15 PM May 24, 2012 10:10:15 PM Individual
10081 May 7, 2012 11:53:32 AM May 7, 2012 11:53:32 AM Individual
10061 May 7, 2012 11:51:30 AM May 7, 2012 11:51:30 AM Individual
Delete
Compare
Rollback
Refresh
Total Rows: 8 (selected: 1)

6. Changes can be compared to one another, in a standard diff format, by selecting them from the list and clicking the **`Compare`** button.



7. The current, live version of the configuration can be reverted to *any* previous version by selecting the desired previous version in the list and clicking the **`Rollback`** button.



Chapter 7. Creating JBoss EAP 6 Resources

7.1. Tracking the Child History

Every **Inventory** tab has a **Child History** subtab. The history lists when child resources have been added or deleted, along with the date when the action occurred and which JBoss ON user initiated it.

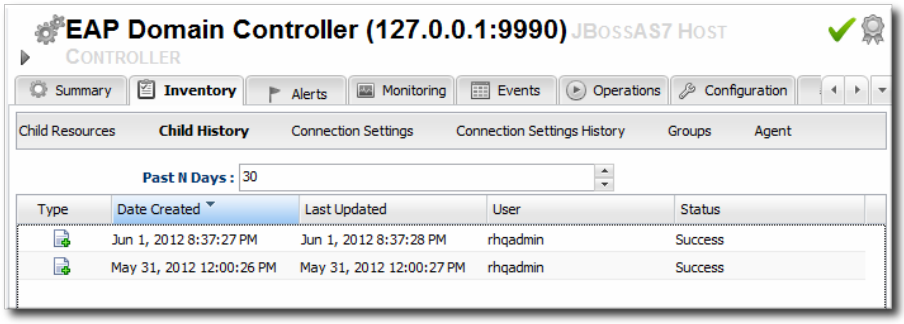


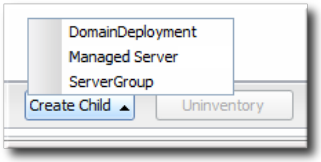
Figure 7.1. Child History

Adding or removing resources through JBoss ON helps maintain that trail of inventory changes.

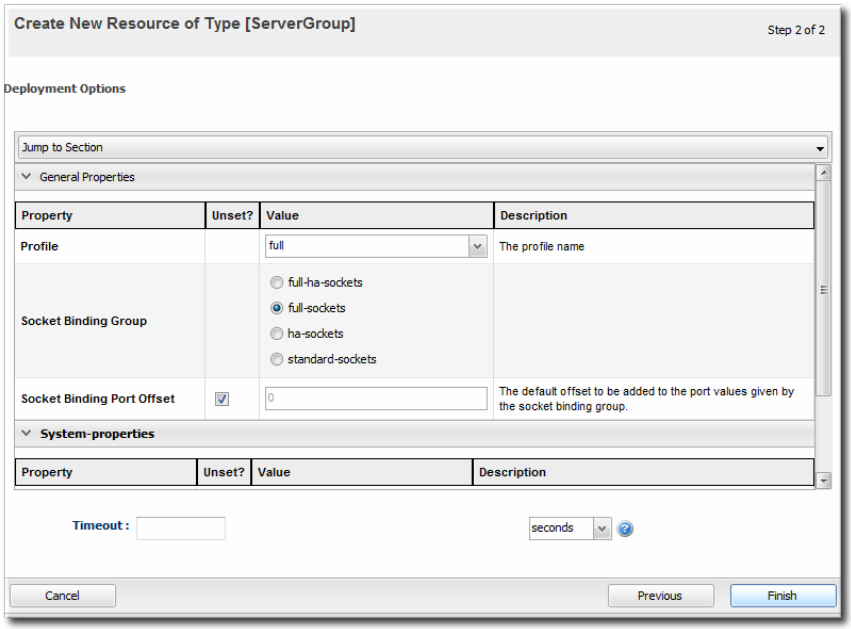
7.2. Creating Server Groups

Server groups are created as part of the domain setup to help manage configuration and content for managed servers.

1. Click the **Inventory** tab in the top menu.
2. Select **Servers - Top Level Imports** in the **Resources** menu table on the left.
3. Right-click the domain controller entry.



4. In the **Create New** menu, select the item for **Server Group**.
5. Enter the name for the new server group.
6. Enter the settings for the server group: the profile to use, the socket bindings group to use, and any system properties to set.



7. Click **Finish**.

7.3. Creating Managed Servers

Managed servers are the functional servers in a domain; they perform the actual client interactions. The managed servers are children of the domain controller; the configuration and content for the managed servers is located in the domain controller configuration and is assigned by server group.

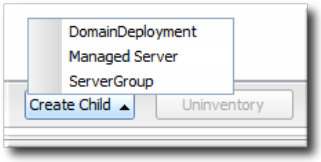
NOTE

Because of the way that domain components can be arranged on physical systems (Section 4.2, "Separating Configuration and Real-Time Operations: Domains"), the managed server may not show up in the same configuration entry as the domain controller which is its parent. The new managed server is listed as a child of the host controller it is created on. The new managed server will show up in the platform (agent) inventory of that host controller.

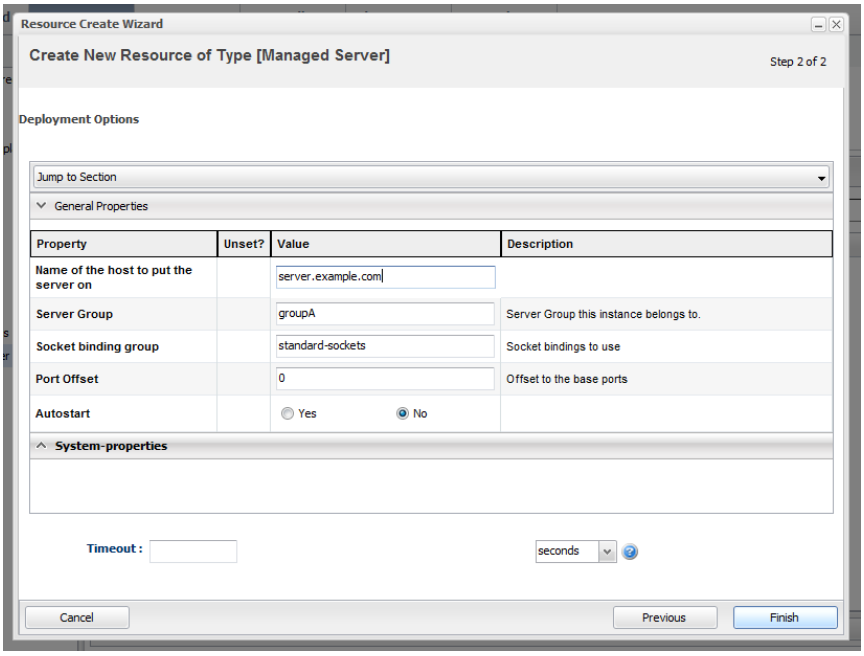
NOTE

The creation process must be performed on the host controller after the job is submitted, and then the local agent must discover the new instances. Depending on where in the schedule the job runs and is completed, it can take several minutes for the new managed server to show up in the agent's discovery queue.

1. Click the **Inventory** tab in the top menu.
2. Select **Servers - Top Level Imports** in the **Resources** menu table on the left.
3. Right-click the domain controller entry.



4. In the **Create New** menu, select the item for **Managed Server**.
5. Enter the name for the new server.
6. Enter the settings for the server:
   - The EAP 6 domain host to create the server on.
   - The server group to add the server to.
   - The socket binding group to use. This gives the base port numbers for the server instance to use.
   - The port offset. This is the number to add to the socket bindings settings to determine the actual port numbers for the server instance. If the socket binding port is 1000, and the offset is 150, then the port number used for that interface on the managed server is 1150.
   - Whether the server starts automatically when the EAP 6 host starts.
   - Any system properties for the server.



7. Click **Finish**.

7.4. Changing JVM Definitions

Managing the JVM settings for instances can help improve performance and better allocate system

resources. Because of the nature of the EAP 6 domain structure, JVM settings work in a kind of cascade: the host controller JVM settings apply to the server group and all servers; any server group JVM settings override the host controller settings and apply to all their managed servers. Then, a JVM definition can be created for a managed resource, and that definition trumps any higher-level JVM settings.

7.4.1. JVM Definitions as Resources

In the EAP 6 management console, the JVM settings are treated as configuration options for server entries. Each server has a configuration tab for JVM configuration, and then there is a separate **JVM Configurations** under the **Server** page for the host controller.

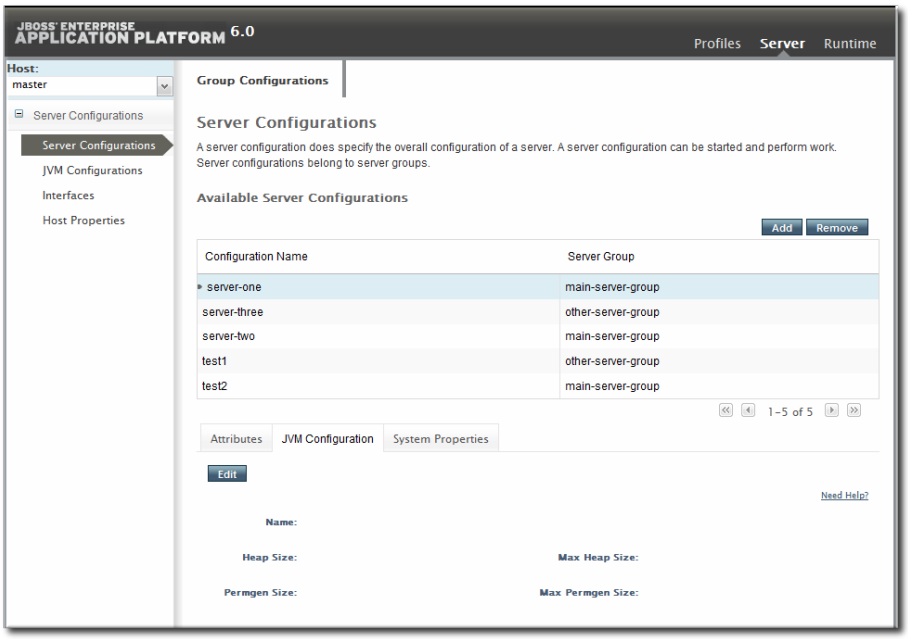


Figure 7.2. JVM Definitions in the EAP 6 Console

As with the managed server configuration, the server group configuration has a tab for JVM settings.

In JBoss ON , JVMs have historically been separate resource types, with their own configuration and monitoring. For EAP 6, JVM definitions are also treated as separate resource types, children of the server or server group they apply to, even though in a structure sense the JVM *definition* is a collection of configuration properties; the JVM itself is the parent resource.

7.4.2. Creating a JVM Definition

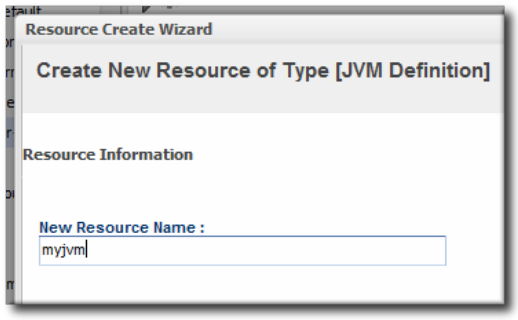
A default JVM definition is already defined for the host controller, and that JVM definition is applied to every server group and, subsequently, every managed server associated with that group.

New JVM definitions can be created for server groups or managed servers to override those default settings. The managed server and server group definitions are based on the host controller JVM definition. Additional JVMs can be added to the host controller to provide a different base to use for new server group and server JVMs.

IMPORTANT

A host controller can have multiple JVM definitions; the default one is used for all managed servers and server groups, while any additional JVM definitions are available as the base definition for custom JVM definitions at the managed server and server group levels. However, a managed server and a server group can have only one JVM definition. The UI will allow multiple JVM definitions to be created for a managed server and a server group, but the final creation step fails if a JVM definition already exists.

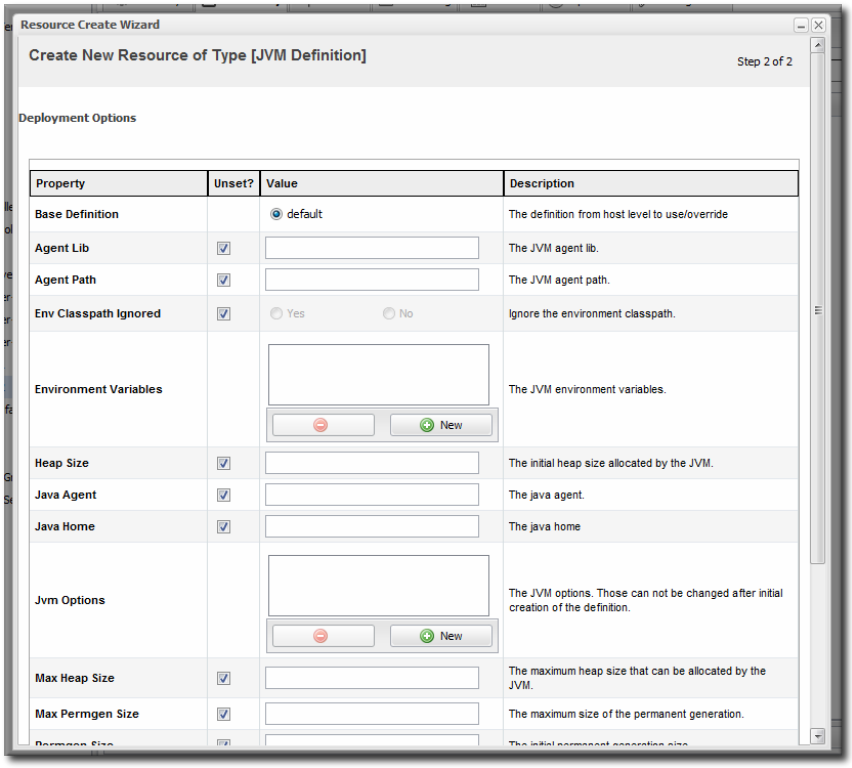
1. Click the **Inventory** tab in the top menu.
2. Select **Servers - Top Level Imports** in the **Resources** menu table on the left.
3. Right-click the entry to which to add the JVM definition.
4. In the **Create New** menu, select the item for **JVM Definition**.
5. Enter the name for the definition.



NOTE

The name of JVM definition can be anything for a host controller. For a managed server or server group, the name of the JVM definition must be the same as the host controller JVM definition which is the base of the definition.

6. Enter the settings for the JVM. Any settings which are left blank use the values defined in the parent JVM definition.



Most of the configuration for the JVM relates to memory and resource usage, along with options to pass environment variables or other settings to the JVM. The values of these settings can have a positive impact on both the resource performance and the overall system performance.

The configuration settings are listed in Table 7.1, “JVM Definition Properties”.

7. Click **`Finish`**.

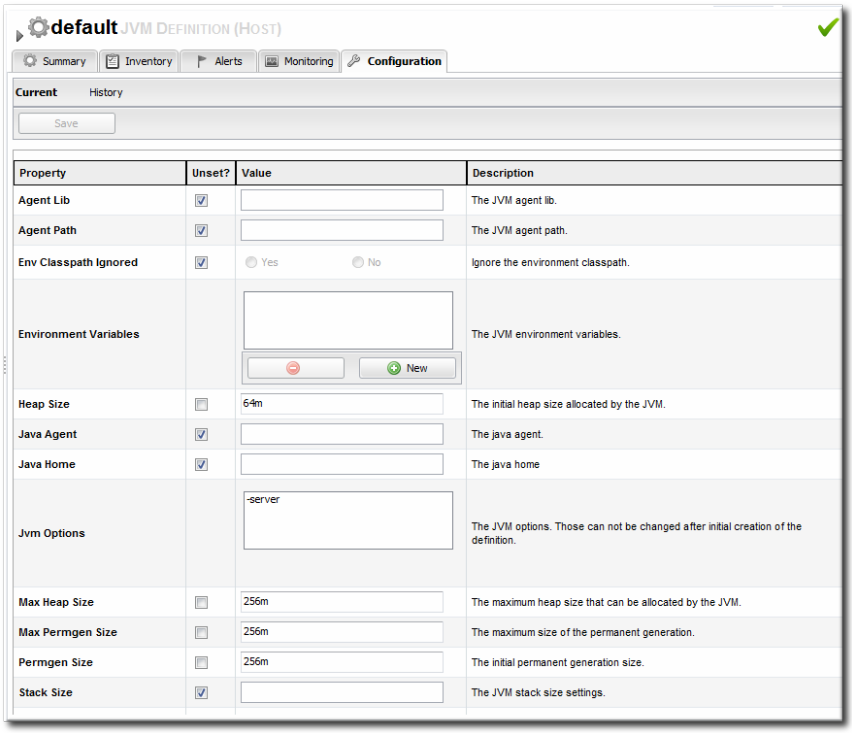
Table 7.1. JVM Definition Properties

Property	Description
JVM Options	Sets any other Java option. Many of these are documented with the Java provider, such as Sun's documentation at http://www.oracle.com/technetwork/java/javase/tech/vmoptions-jsp-140102.html.
Environment variables	Sets environments variables to use with the JVM.
Heap Size	Sets the initial heap size for the JVM.
Max Heap Size	Sets the maximum allowed heap size for the JVM. Setting this too low can cause out of memory errors.
Permgen Size	Sets the initial size of the permanent generation.
Max Permgen Size	Sets the maximum size of the permanent generation.
Type	The type of JVM. This can be Sun or IBM, the two supported Java types for JBoss ON.

7.4.3. Editing a JVM Definition

The JVM definition is edited by editing the configuration properties for the JVM definition resource.

1. Click the **`Inventory`** tab in the top menu.
2. Select **`Servers - Top Level Imports`** in the **`Resources`** menu table on the left.
3. Select the EAP server and navigate to the appropriate JVM definition entry.
4. Open the **`Configuration`** tab.



5. Change any of the JVM settings. These are listed in Table 7.1, "JVM Definition Properties".
6. Click the **`Save`** button at the top of the page.

7.5. A Short List of Parent-Child Resources

There are hundreds of resource types defined in the JBoss EAP 6 agent plug-in, covering all major servers, subsystem types, and services. The Resource Reference covers all of the different resource types and lists child types for each.

Table 7.2, "A Short List of Parent-Child Resources" lists some common JBoss EAP 6 resource types and what children resource can be created for them through the JBoss ON UI.

NOTE

Creating a child resource must be performed on the parent resource after the job is submitted. Then the local agent must discover the new instances.
Depending on where in the schedule the job runs and is completed, it can take several minutes for the new managed server to show up in the agent's discovery queue.

Table 7.2. A Short List of Parent-Child Resources

Resource	Child Resource Types
Standalone server	Deployment
Domain Controller	Domain deployment Managed servers Server groups
Host	JVM definition
Server Groups	Deployment JVM definition
Datasources (under Profiles)	Datasource XA Datasource
Infinispan > Hibernate	Cache
Logging	Async handler Console handler Custom handler File handler Logger Periodic Rotating File handler Size Rotating File handler
Web	Connector Vhost

Chapter 8. Deploying Web Applications

Web applications are a unique type of resource. They are added as child resources of servers, so they have a place within the inventory with their own entries, possibly even their own child resources, and independent configuration for monitoring, alerting, and operations. However, web applications are *content-backed resources*. Ultimately, they are software packages, not true servers or services.

Managing a content-backed resource requires a path to make the initial deployment (creating the deployment as a child resource) and also a path for patches and updates (content updates).

8.1. Runtime Information and Deployment Resources

8.1.1. Views of Deployments

Both EAP 6 and JBoss ON offer clear workflows for adding web applications to a central content repository and then associating that content with a server group. But the perspectives they offer are very different, so deciding which console to use depends on what needs to be accomplished at that particular moment.

The EAP 6 console is focused on immediately associating content with a server group. In the EAP 6 console, deployments are treated similarly to shared configuration, like a JVM definition. A content package is deployed to a domain controller or a server group and is then associated with a managed server.

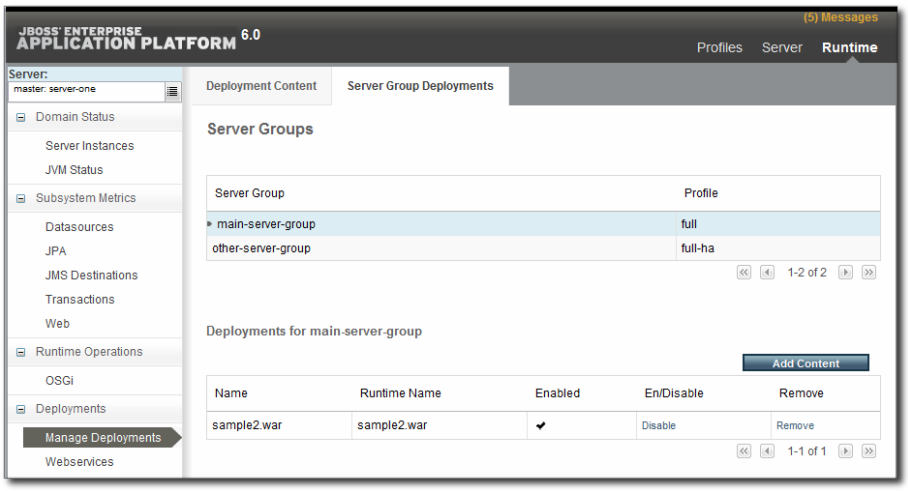


Figure 8.1. Deployments in the Runtime Page

JBoss ON is focused on lifecycle management for web application, an historical and changing perspective. There are a few fundamental differences in how JBoss ON defines web applications:

- A web application is treated as a separate entity, in and of itself. It has its own place in the inventory; its association with domains and server groups is reflected as it is a child of those resources.

  JBoss ON also records package details, like its file size and an identifying SHA256 hash.
- A web application has a history. Updates to packages are recorded in a changelog which makes it possible to track how the application has been maintained.
- Web applications can be monitored. Response time metrics specifically track the performance of web applications, apart from the performance or monitoring of the underlying server.

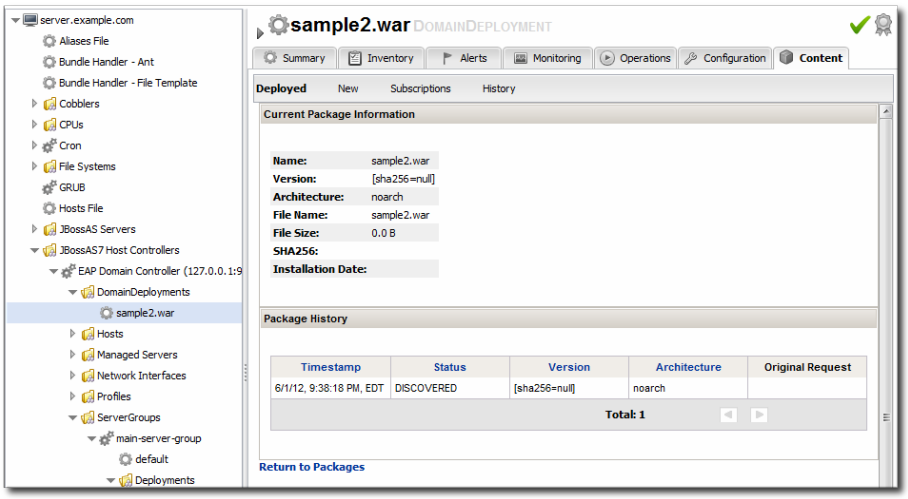


Figure 8.2. Deployment Resource Details in JBoss ON

The reason for this emphasis on web applications as a resource is that the core purpose is not the simple task of deploying content to a server. The EAP 6 console does that very simply and cleanly. The purpose of JBoss ON management is administration of that content of a web application:

- Creating content repositories to store patches and updates
- Tracking multiple versions of content
- Reverting software packages to a previous version (particularly for standalone servers)
- Using the same content repository for multiple EAP instances, including multiple domains and standalone server
- Tracking (and auditing) changes to content

8.1.2. Deployment Paths for Standalone Servers and Domains

There is a radically different structure between standalone servers and domains, and that impacts the content flow for deploying web applications.

Web application deployments in standalone servers are similar to EAP 5, in that there is a local filesystem directory which is used for deployments. Locally, applications can be added to that deployment directory and hot-deployed. Using JBoss ON, there are two paths for deploying web applications:

- Create a deployment as a child resource
- Use a bundle to provision the application in the deployment directory

Using bundles is a versatile and practical management method, because the bundle system has capabilities to store multiple versions, rollback updates, skip versions for updates, or purge deployments, all with a single button.

Unfortunately, the bundles system only works with real filesystem locations. The distributed and modular structure for JBoss EAP 6 domains means that bundles are not an option.

For EAP 6 domains, the content is deployed as a child resource to the domain and then is propagated to server groups by administrators.

It lacks the flexibility and finesse of the bundles system, but using the ordinary content system still offers ways to control content versions and updates. Content-backed resources can be associated with content repositories in JBoss ON. These repositories can store all sorts of software packages, from different content sources. Packages from the repo can be installed on the resource selectively. Additionally, a resource can be associated with multiple repos. While not as fluid as updating or reverting changes with bundles, the content repositories in JBoss ON provide a pathway to apply individual patch files to a web application or to roll out full updates.

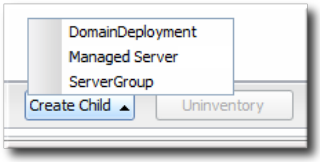
Content management in general is described in "Deploying Applications and Content". This covers setting up content repositories, subscribing resources, and pushing updates.

8.2. Deploying Web Applications to a Domain

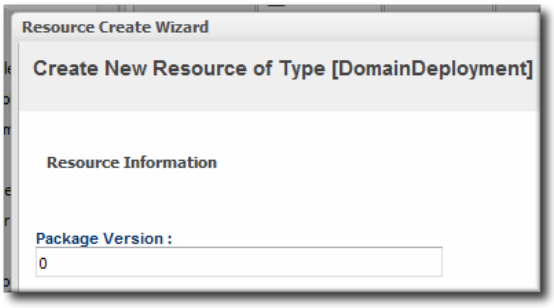
NOTE

As with creating other child resources, it can take several minutes for the new deployment to be added and visible in the JBoss ON inventory because the new resource has to be created on the local system and then discovered by the agent. If the discovery scan is running when the resource is created, then it may take until the next discovery scan to be detected, as long as 15 minutes.

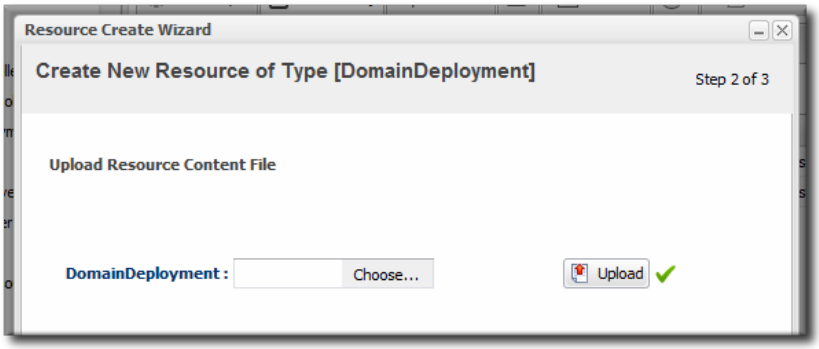
1. Click the **Inventory** tab in the top menu.
2. Select **Servers - Top Level Imports** in the **Resources** menu table on the left. Select the JBoss EAP 6 domain controller.
3. Right-click the domain controller entry.



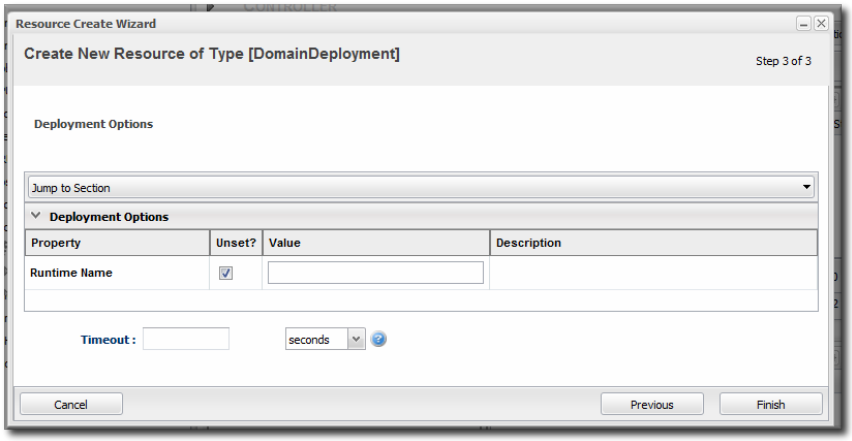
4. In the **Create New** menu, select the item for **DomainDeployment**.
5. Enter the version number.



6. Upload the EAR, WAR, or JAR file.



7. Optionally, enter a runtime name for the deployment.



8. At the bottom of the wizard, set an optional timeout period. This is how long the JBoss ON server will wait during the deployment process before determining that the deployment has failed.

NOTE

The timeout period only applies to the server's reporting a result. If the operation continues running, it can still complete successfully, and the web application is deployed.
Particularly for large application files, do not set a low timeout period, or the server will mark the deployment as having failed. If the deployment completes later, the web application must be imported into the inventory manually; it will not be discovered by the agent.

9. Click **`Finish`**.

Once the package is uploaded, a new resource entry is added to the **`DomainDeployments`** directory for the domain.

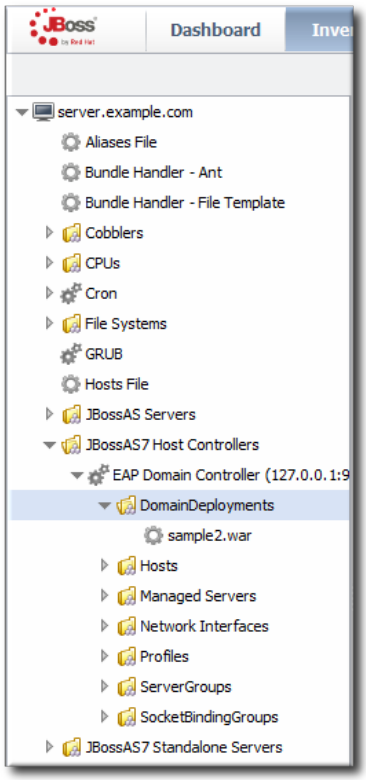


Figure 8.3. Domain Deployments Directory

8.3. Assigning Web Applications to a Server Group

Assigning web applications to different groups manually allows administrators to control the lifecycle for applications, as expanded in Section 8.4, "*Extended Example:* Assigning Web Applications and Managing Updates". The same content is used in different operating environments.

TIP

Web applications can be deployed directly to a server group, much like the process to deploying it to a domain, in Section 8.2, "Deploying Web Applications to a Domain".
The deployment path for a domain is to upload the content to the domain controller and then to distribute that content to a server group. Creating a deployment on a server group follows the same process; it deploys it to the domain and then sends it to the server group. It just provides a shortcut.
If the content already exists on the domain controller, then the content must be deployed through the domain controller operation described here.

Deploying the content to a domain first uses the domain as a kind of repository. From the domain, the same package can be deployed to multiple groups. Additionally, deploying to a domain allows administrators to deploy multiple packages in the same operation to a group, either simultaneously or in a specified order. This can help control content streams to servers.

IMPORTANT

Newly-deployed content for a server group may not show up in the JBoss ON inventory for as long as 24 hours, even if it was successfully created. By default, discovery scans for services are only made every 24 hours. To see the new content-backed resource, run the agent's discovery scan and manually discover it.

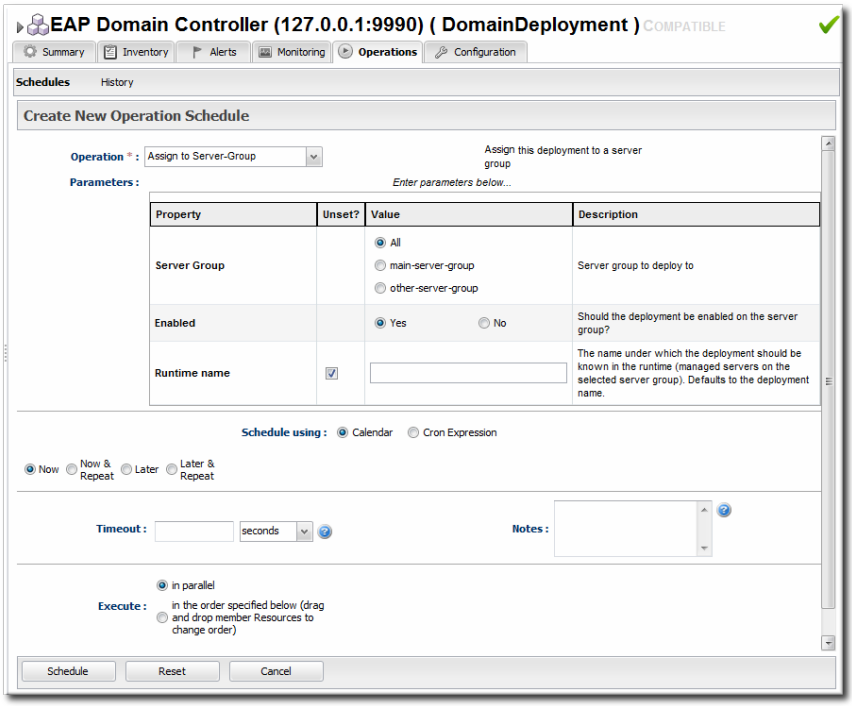
NOTE

The deployment is created in the domain as a child resource, then assigned to a server group as an operation, and finally updated through the domain's **Content** tab.

1. Click the **Inventory** tab in the top menu.
2. Select **Servers - Top Level Imports** in the **Resources** menu table on the left. Select the JBoss EAP 6 domain controller.
3. Expand the **DomainDeployments** folder.

   To deploy all web applications in the deployments folder, select the deployments folder itself to run the operation.

   To deploy a single web application, select the specific web application.
4. Open the **Operations** tab for the server.
5. Click the **New** button at the bottom of the page.
6. Select the **Assign to Server-Group** option from the drop-down menu.



7. Select which group (or all groups) to deploy the application to.
8. Enter the standard information for operations, like the schedule for when to run the operation and the timeout period.

   NOTE

   The timeout period sets how long the server waits before assuming that the operation has timed out and failed. This does not necessarily mean that the operation has failed or stopped running; it could complete successfully past the time out period.

9. If there are multiple web applications being deployed, then set the **Execute** radio button to the way set the order that the packages are deployed. All packages can be deployed at once, or they can be deployed in a specific order.
10. Click the **Schedule** button.
11. Optionally, run a discovery scan on the agent to discover the new content resource. By default, discovery scans for services are only made every 24 hours, so there could be a long delay in discovering new content.
    a. Open the agent entry in the UI.
    b. Open the **Operations** tab, and select the *execute prompt command* operation.

c. Enter the **`discovery`** command as an operation rgument. This runs a discovery scan.
d. Click the **`Schedule`** button.

8.4. *Extended Example:* Assigning Web Applications and Managing Updates

The Setup

Tim the IT Guy wants to have a clear progression for web applications, from development through staging and production. The native structure in EAP 6 allows him to create different server groups and deploy content from his central domain controller to the appropriate server groups as it passes testing at each stage.

What Tim the IT Guy wants is to add a layer on top of that path that allows him to create an apply patches and updates. He has to be able to manage fixes and to audit when they were applied as part of his maintenance schedule.

The Plan

1. Tim first outlines what server groups he needs to maintain. For a simple environment, he just wants three groups: testing, staging, and production.
2. He creates two content repositories, one for patches and one for new versions of the web application.
3. He creates the domain deployment and then promotes the web application to the testing server group.
4. Tim configures response time monitoring for the web application. Once it meets the required performance parameters in the testing area, Tim promotes the deployment to the staging and then production server groups.

The Result

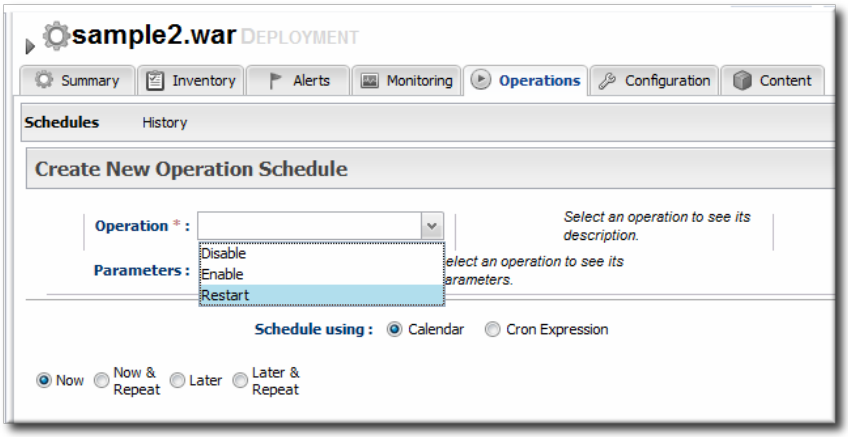
The package history for each deployment allows Tim to track when the web application was deployed, its version, and its content.

Tim can apply patches or full updates by deploying content to his repositories and installing it on every subscribed resource. These package changes, including new version numbers for the content, are included in the package history.

Because the patch and update repositories are configured in JBoss ON itself, not a specific JBoss EAP 6 domain, the packages there can be used with other domains and standalone servers, if need be.

8.5. Enabling and Disabling Web Applications

1. Click the **`Inventory`** tab in the top menu.
2. Select **`Servers - Top Level Imports`** in the **`Resources`** menu table on the left. Select the JBoss EAP 6 domain controller.
3. In the inventory tree, expand the **`ServerGroups`** folder, and select the server group from the list.
4. Expand the **`Deployments`** folder, and select the web application to enable or disable.
5. Open the **`Operations`** tab for the web app.
6. Click the **`New`** button at the bottom of the page.
7. Select the **`Enable`** (or **`Disable`**) option from the drop-down menu.

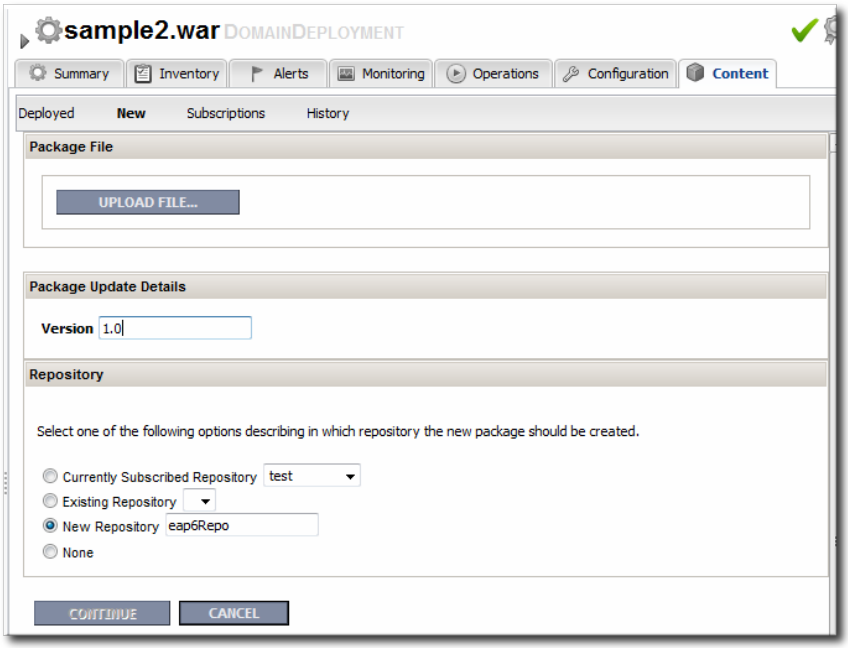


8. Click the **`Schedule`** button.

8.6. Updating Deployment Content

1. Click the **`Inventory`** tab in the top menu.

2. Select **Servers - Top Level Imports** in the **Resources** menu table on the left. Select the JBoss EAP 6 server.
3. Open the **Deployments** folder (for the domain, standalone server, or server group), and select the web application to update.
4. On the web application details page, open the **Content** tab, and click the **New** subtab.
5. Click the **UPLOAD NEW PACKAGE** button.
6. Click the **UPLOAD FILE** button.
7. In the pop-up window, click the **Add** button, and browse the local filesystem to the updated content file to be uploaded.
8. Click the **UPLOAD** button.
9. Select the repository where the web application package should be stored. While this is not required, it is beneficial to store the updated package in JBoss ON so that it is available to other JBoss EAP 6 deployments.



10. Fill in the version information.
11. Confirm the details for the new package, then click **CONTINUE**.

When the package is successfully uploaded, the UI redirects to the history page on the **Content** tab.

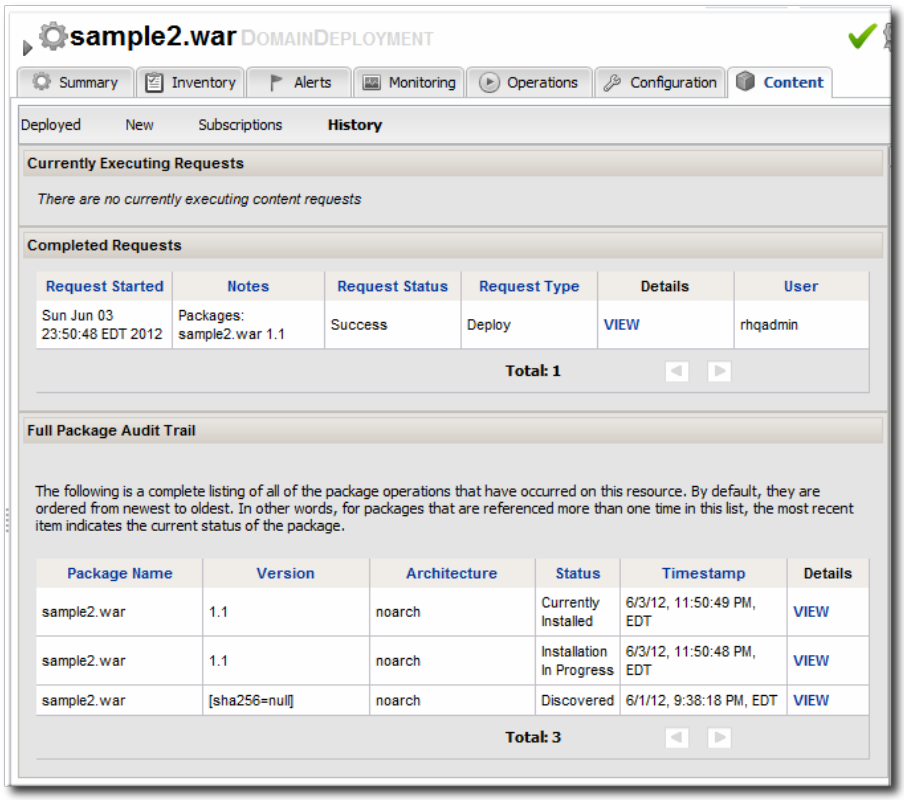


Figure 8.4. Deployment History for a Resource

8.7. Deploying Web Applications to a Standalone Server

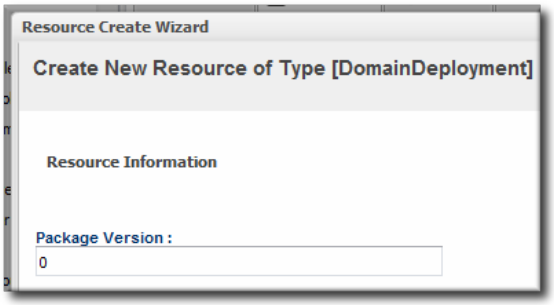
The standalone server uses a specific, filesystem directory as a deployment directory. As with domain deployments, content can be deployed by creating a child resource. However, because of the simpler structure of the standalone server, content can also be deployed through content bundles. Bundles provide simplified update and revert paths, as well as a clearer versioning system.

8.7.1. Deploying a Web Application as a Child Resource

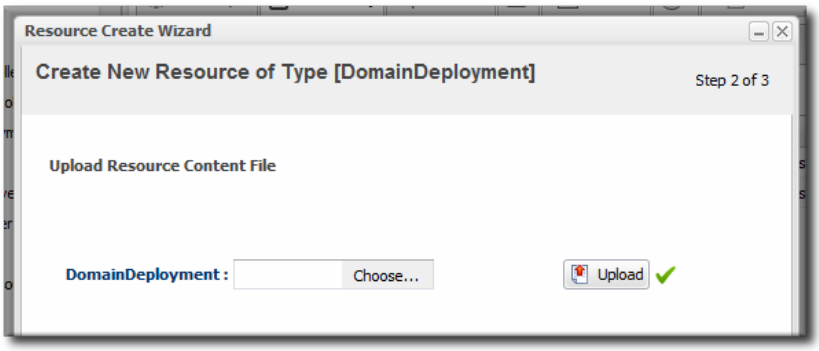
IMPORTANT

New deployments may not show up in the JBoss ON inventory for a few minutes or for as long as 24 hours, even if it was successfully created. This is because the new resource must be discovered by the agent after it is created for it to be added to the inventory. By default, discovery scans for services are only made every 24 hours.
To see it immediately, run an *execute prompt command* operation on the agent and enter the **`discovery`** command. This runs a discovery scan.

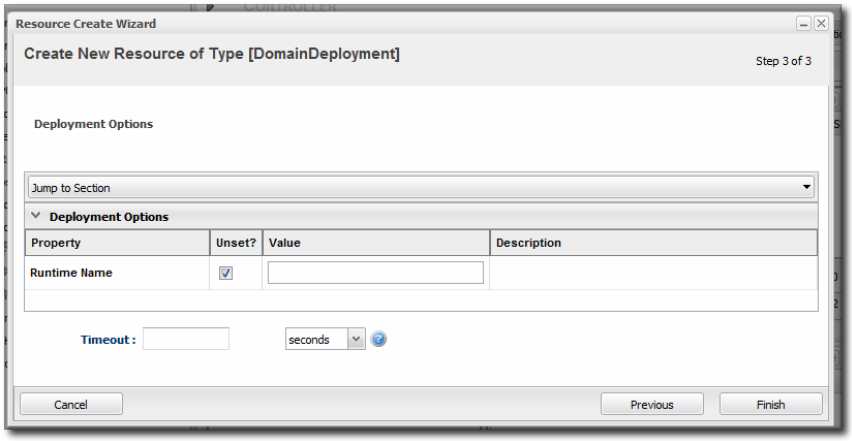
1. Click the **`Inventory`** tab in the top menu.
2. Select **`Servers - Top Level Imports`** in the **`Resources`** menu table on the left. Select the JBoss EAP 6 standalone server.
3. Right-click the standalone server entry in the navigation tree.
4. In the **`Create New`** menu, select the item for **`Deployment`**.
5. Enter the version number.



6. Upload the EAR, WAR, or JAR file.



7. Optionally, enter a runtime name for the deployment.



8. At the bottom of the wizard, set an optional timeout period. This is how long the JBoss ON server will wait during the deployment process before determining that the deployment has failed.

NOTE

The timeout period only applies to the server's reporting a result. If the operation continues running, it can still complete successfully, and the web application is deployed. Particularly for large application files, do not set a low timeout period, or the server will mark the deployment as having failed. If the deployment completes later, the web application must be imported into the inventory manually; it will not be discovered by the agent.

9. Click **`Finish`**.

8.7.2. Deploying Web Applications Through Bundles

Creating bundle archives and recipes, defining destinations, and managing bundle versions are described in "Deploying Applications and Content".

1. In the top menu, click the **`Bundles`** tab.

2. Upload the distribution file, and go through the deployment wizard to create the bundle version.
3. Scroll to the bottom of the window and click the **`Deploy`** button.
4. Select the bundles to deploy.
5. Define the destination information for the JBoss standalone server. A destination is a combination of a compatible group (containing the JBoss resource) and the directory to which to deploy the bundle.
6. Complete the deployment wizard, setting any require properties.

8.8. Tracking Content History and Reverting Changes

The deployment history page tracks changes to a content-backed resource, meaning it lists all update attempts, timestamps, and version numbers.

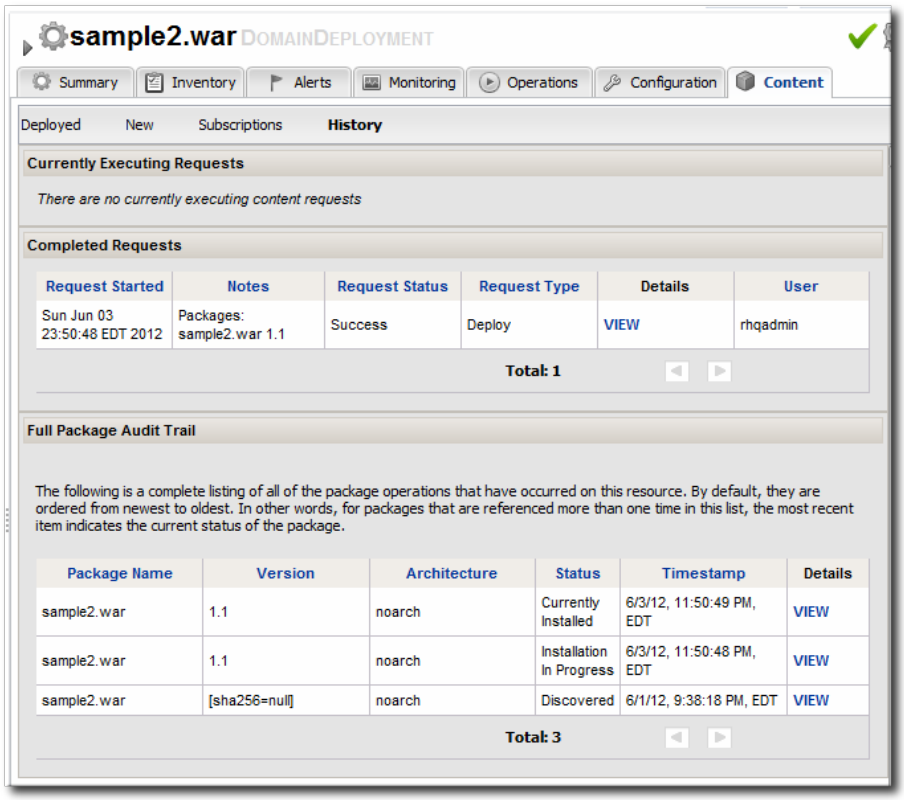


Figure 8.5. Deployment History for a Resource

Web applications deployed to standalone servers through the bundles provisioning system have a straightforward path to revert changes. There is a **`Revert`** button on the details page for the latest deployment which goes back to the immediate past deployed version.

Deployments to a server group or domain, however, do not have the same clear path. Any attempt to revert a content deployment ultimately means to replace it with a different version.

There are options for how to change the package when a direct reversion is not possible:

- If the domain deployment or server group deployment is associated with a content repository, then upload an updated package to the content repository and sync the change over to the associated resources.
- Upload an updated package to the domain deployment and use the *deploy to group* operation to send the updated package to the server groups.

8.9. Troubleshooting Deployments

Q: My deployment says it failed, but when I tried to redeploy the package, it fails because it says the resource already exists.

A: One possible cause is the timeout setting when the package was originally deployed. The timeout period sets how long the JBoss ON server waits before assuming the deploy operation failed — it does not actually stop or limit the deploy operation itself. Even if the operation hits the timeout limit, the operation can still complete successfully, which successfully deploys the web application.

If the operation times out, then the agent will not discover the new resource automatically. The web application must be imported into the inventory manually.

Q: I tried to deploy a package to a server group and it seemed to be successful, but I don't see the deployment listed.

A: There are a couple of possible causes:

- The agent discovery scan hasn't run. There can be several hours between discovery runs, so it can take awhile for the application to appear in the discovery queue. To work around this, run a discovery scan on the agent.
- The package has already been deployed to the *domain*. Creating a deployment child on a server group actually attempts to create the deployment on the domain (where the content is stored) and then deploys it to the server group. If the package already exists in the domain, then attempting to re-create the deployment on the server group fails as a duplicate.

  In this case, use the *deploy to server group* operation on the domain controller to deploy the application.

Chapter 9. Monitoring JBoss EAP 6 Resources

9.1. Runtime Information and JBoss ON Monitoring

Metrics are a live-stream of information about how a resource or group of resources are performing. Metrics are vital signs, like the heart rate, blood pressure, and oxygen levels for a hospital patient. For resources, those vital statistics are elements like availability, page hits for database caches, or active connections for datasources or web applications. Those metrics show resource health, observed at a specific moment.

The JBoss EAP 6 console shows that current and immediate snapshot of information for a handful of critical metrics for a few critical resources, like datasources, transactions, and web applications.

This changing information is displayed in the **`Runtime`** tab of the EAP 6 console.

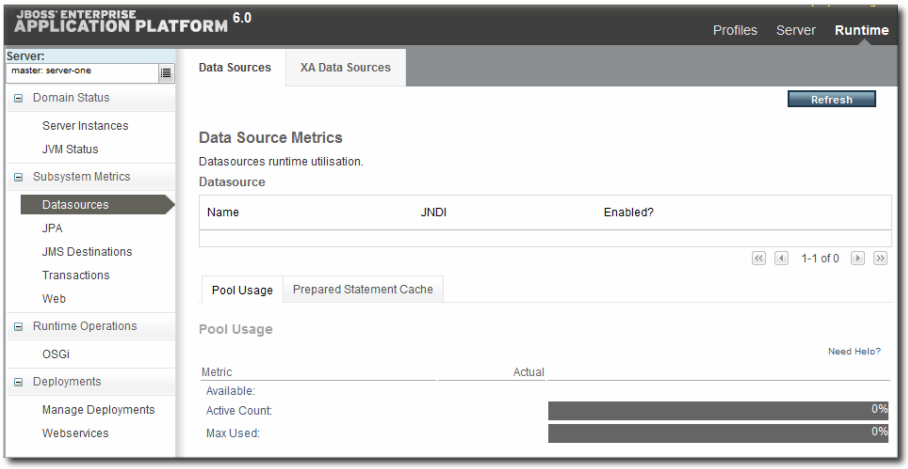


Figure 9.1. EAP 6 Runtime Metrics

Runtime information shows the health for that exact moment in time.

Looking at the runtime information *over time* gives a much better perspective of how a resource is performing, generally. Every resource in JBoss ON has areas to view and configure monitoring and to configure alerting based on that monitoring.

JBoss ON automatically processes runtime metrics in a few different ways, to help administrators process performance data:

- Time-based monitoring graphs for each metric
- Operating parameters, or *baselines* calculated on the real performance of the resource
- Availability uptime percentages, recovery times, and mean down time

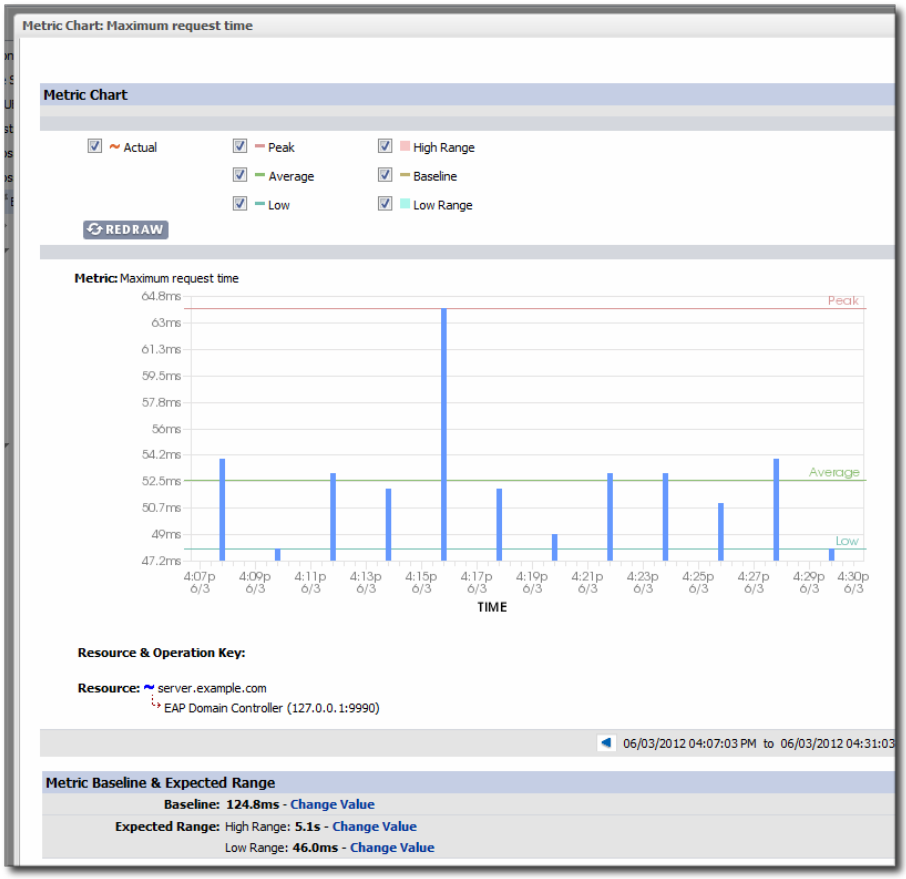


Figure 9.2. JBoss ON Baselines for a Single Metric

JBoss ON also exposes more monitoring data for individual resources through its GUI:

- More metrics are available for resources. At a minimum, every resource has availability monitoring. Some resources — like datasources and managed servers — have dozens more metrics that can be enabled for routine monitoring.
- Response time monitoring can be configured for specific URLs and pages for web applications. This allows administrators to track usability and customer experience for web applications along with internal metrics like connection counts.
- Event log monitoring can filter important messages from different error logs. This allows JBoss ON to respond (through alerts) to issues that may not have crossed a performance threshold yet, and it also makes it possible for administrators to correlate log events with observed performance metrics.

9.2. Configuring Response Time Metrics for JBoss EAP 6/AS 7

JBoss ON can monitor application performance by testing how responsive deployed applications are to client requests. Specifically, JBoss ON can determine how quickly an application responds to requests. This is called a *response time measurement*.

To collect response time metrics for a JBoss EAP 6/AS 7 server, first install the servlet JAR for the response time filter and configure the web server to use it. Then, enable the metrics collection for the web resource.

NOTE

URL, response time, and call-time monitoring is described more in the URL monitoring chapter of "Setting up Monitoring, Alerts, and Operations".

9.2.1. Installing the Response Time Filters

1. Make sure that you have created a management user to access the JBoss EAP 6 instance.
   For more information, see the JBoss AS 7.1 documentation.
2. Download the response time packages for JBoss from the JBoss ON UI. The response time filters are packaged as AS 7 modules. There are two modules to obtain:

```
rhq-rtfilter-module.zip
rhq-rtfilter-subsystem-module.zip
```

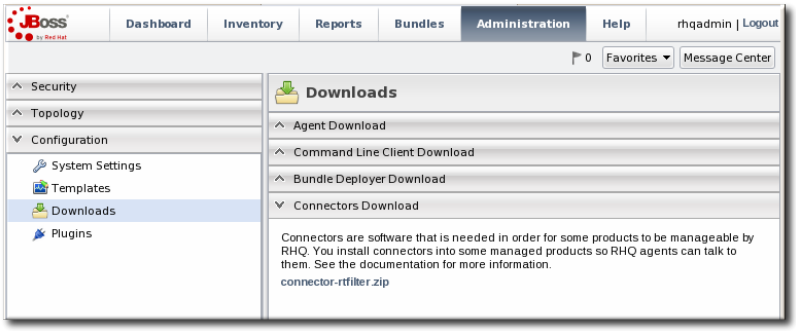
TIP

This can also be done from the command line using **wget**:

```
[root@server ~]# wget
http://server.example.com:7080/downloads/connectors/rhq-rtfilter-
module.zip
[root@server ~]# wget
http://server.example.com:7080/downloads/connectors/rhq-rtfilter-
subsystem-module.zip
```

a. Click the **Administration** tab in the top menu.

b. In the **Configuration** menu box on the left, select the **Downloads** item.



c. Click the **rhq-rtfilter-module.zip** and **rhq-rtfilter-subsystem-module.zip** links, and save the files to an accessible directory, like the **/tmp** directory.

3. Open the **modules/** directory for the JBoss EAP 6 instance. For example:

```
[root@server ~]# cd /opt/jboss-eap-6.0/modules/
```

4. Unzip the **rhq-rtfilter-module.zip** archive to install the response time filter JAR and the associated **module.xml** file.

```
[root@server modules]# unzip /tmp/rhq-rtfilter-module.zip
```

5. Open the configuration file for the server, **domain.xml** or **standalone.xml**.

6. Deploy the response time module globally by adding the module to the list of global modules in the **<subsystem>** element.

```
<profile...
 <subsystem xmlns="urn:jboss:domain:ee:1.1">
  <global-modules>
      <module name="org.rhq.helpers.rhq-rtfilter" slot="main"/>
  </global-modules>
 </subsystem>
</profile>
```

7. Save the file.

8. Unzip the **rhq-rtfilter-subsystem-module.zip** archive to install the subsystem response time filter JAR and the associated **module.xml** file.

```
[root@server modules]# unzip /tmp/rhq-rtfilter-subsystem-module.zip
```

This installs the filters as a subsystem for the application server or individual web apps.

9. After the filters have been installed, the JBoss EAP 6 server needs to be configured to use them.

The response time filter can be deployed globally, for all web applications hosted by the EAP/AS instance, or it can be configured for a specific web application.

To deploy the filter as a global subsystem:

a. Open the configuration file for the server, **domain.xml** or **standalone.xml**.

b. Add the an **<extensions>** element for the response time filter.

```
<extension module="org.rhq.helpers.rhq-rtfilter-subsystem"/>
```

c. Add a **<subsystem>** element beneath the **<profile** element.

All that is required for response time filtering to work is the default **<subsystem>** element, without any optional parameters. However, the parameters can be uncommented and set as necessary; the different ones are described in Table 9.1, “Parameters Available for User-Defined <filter> Settings”.

The **<subsystem>** element should be added even if none of the optional parameters are set.

TIP

Instead of setting an optional parameters in the configuration file, they can be set in the configuration settings for the web runtime resource through the JBoss ON GUI.

```
<subsystem xmlns="urn:rhq:rtfilter:1.0">
    <!-- Optional parameters.

       <init-param>
           <param-name>chopQueryString</param-name>
           <param-value>true</param-value>
       </init-param>
       <init-param>
           <param-name>logDirectory</param-name>
          <param-value>/tmp</param-value>
       </init-param>
       <init-param>
           <param-name>logFilePrefix</param-name>
           <param-value>localhost_7080_</param-value>
       </init-param>
       <init-param>
           <param-name>dontLogRegEx</param-name>
           <param-value></param-value>
       </init-param>
       <init-param>
          <param-name>matchOnUriOnly</param-name>
          <param-value>true</param-value>
       </init-param>
       <init-param>
           <param-name>timeBetweenFlushesInSec</param-name>
           <param-value>73</param-value>
       </init-param>
       <init-param>
           <param-name>flushAfterLines</param-name>
           <param-value>13</param-value>
       </init-param>
       <init-param>
           <param-name>maxLogFileSize</param-name>
           <param-value>5242880</param-value>
       </init-param>
-->
</subsystem>
```

To configure the response time filters for an individual web application:

a. Open the web application's **web.xml** file.

```
[root@server ~]# vim WARHomeDir/WEB-INF/web.xml
```

b. Add the filter and, depending on the configuration, filter mapping elements to the file. This activates the response time filtering.

All that is required for response time filtering to work is the default **<filter>** element, without any optional parameters. However, the parameters can be uncommented and set as necessary; the different ones are described in Table 9.1, “Parameters Available for User-Defined <filter> Settings”.

TIP

Instead of setting an optional parameters in the configuration file, they can be set in the configuration settings for the resource through the JBoss ON GUI.

```
    <filter>
        <filter-name>RhqRtFilter</filter-name>
        <filter-class>org.rhq.helpers.rtfilter.filter.RtFilter</filter-
class>

<!-- Optional parameters.

        <init-param>
            <param-name>chopQueryString</param-name>
            <param-value>true</param-value>
        </init-param>
        <init-param>
            <param-name>logDirectory</param-name>
           <param-value>/tmp</param-value>
        </init-param>
        <init-param>
            <param-name>logFilePrefix</param-name>
            <param-value>localhost_7080_</param-value>
        </init-param>
        <init-param>
            <param-name>dontLogRegEx</param-name>
            <param-value></param-value>
        </init-param>
        <init-param>
           <param-name>matchOnUriOnly</param-name>
           <param-value>true</param-value>
        </init-param>
        <init-param>
            <param-name>timeBetweenFlushesInSec</param-name>
            <param-value>73</param-value>
        </init-param>
        <init-param>
            <param-name>flushAfterLines</param-name>
            <param-value>13</param-value>
        </init-param>
        <init-param>
            <param-name>maxLogFileSize</param-name>
            <param-value>5242880</param-value>
        </init-param>
-->

    </filter>

    <!-- Use this only when also enabling the RhqRtFilter in the filter
    <filter-mapping>
        <filter-name>RhqRtFilter</filter-name>
        <url-pattern>/*</url-pattern>
    </filter-mapping>
    -->
```

10. Restart the JBoss EAP/AS server to load the new **`web.xml`** settings.

Table 9.1. Parameters Available for User-Defined <filter> Settings

Parameter	Description
chopQueryString	Only the URI part of a query will be logged if this parameter is set to true. Otherwise the whole query line will be logged. Default is true.
logDirectory	The directory where the log files will be written to. Default setting is {**jboss.server.log.dir}/rt/** (usually **server/xxx/log/rt**). If this property is not defined, the fallback is {**java.io.tmpdir}/rt/** (**/tmp/** on UNIX®, and **~/Application Data/Local Settings/Temp** – check the **TEMP** environment variable) is used. If you specify this init parameter, no directory **rt/** will be created, but the directory you have provided will be taken literally.
logFilePrefix	A prefix that is put in front of the log file names. Default is the empty string.
dontLogRegEx	A regular expression that is applied to query strings. See java.util.regex.Pattern. If the parameter is not given or an empty string, no pattern is applied.
matchOnUriOnly	Should the dontLogRegEx be applied to the URI part of the query (true) or to the whole query string (false). Default is true.
timeBetweenFlushesInSec	Log lines are buffered by default. When the given number of seconds have passed and a new request is received, the buffered lines will be flushed to disk even if the number of lines to flush after (see next point) is not yet reached.. Default value is 60 seconds (1 Minute).
flushAfterLines	Log lines are buffered by default. When the given number of lines have been buffered, they are flushed to disk. Default value is 10 lines.
maxLogFileSize	The maximum allowed size, in bytes, of the log files; if a log file exceeds this limit, the filter will truncate it; the default value is 5242880 (5 MB).

vHostMappingFile	This properties file must exist on the Tomcat process classpath. For example, in the ../conf/vhost-mappings.properties. The file contains mappings from the 'incoming' vhost (server name) to the vhost that should be used as the prefix in the response time log file name. If no mapping is present (no file or no entry response times are set), then the incoming vhost (server name) is used. For example:
	`pickeldi.users.acme.com=pickeldi`
	`pickeldi=`
	`%HOST%=`
	The first mapping states that if the incoming vhost is 'host1.users.acme.com', then the log file name should get a vhost of 'host1' as prefix, separated by a _ from the context root portion. The second mapping states that if the 'incoming' vhost is 'host1', then no prefix, and no _, should be used. The third mapping uses a special left-hand-side token, '%HOST%'. This mapping states that if the 'incoming' vhost is a representation of localhost then no prefix, and no _ , should be used.
	%HOST% will match the host name, or canonical host name or IP address, as returned by the implementation of InetAddress.getLocalHost().
	The second and third mappings are examples of empty right hand side, but could just as well have provided a vhost.
	This is a one time replacement. There is no recursion in the form that the result of the first line would then be applied to the second one.

9.2.2. Enabling the Call-Time Metric

Response time metrics are configured on a live application deployment. A deployment resource is a child of either a standalone EAP 6 server or of a server group.

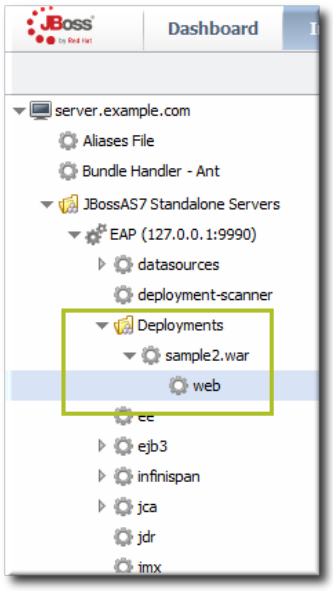
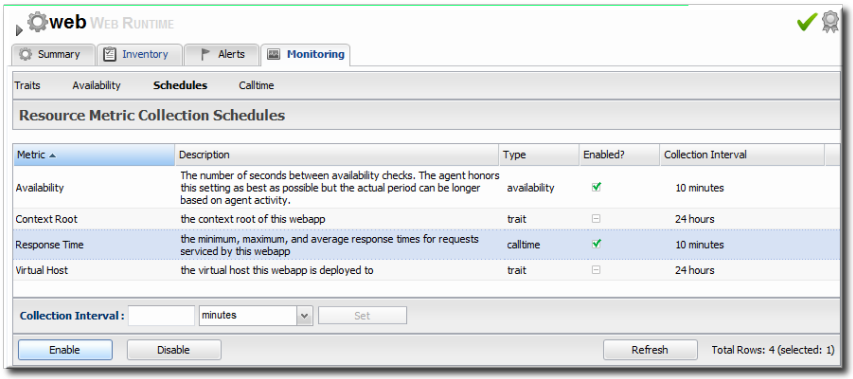
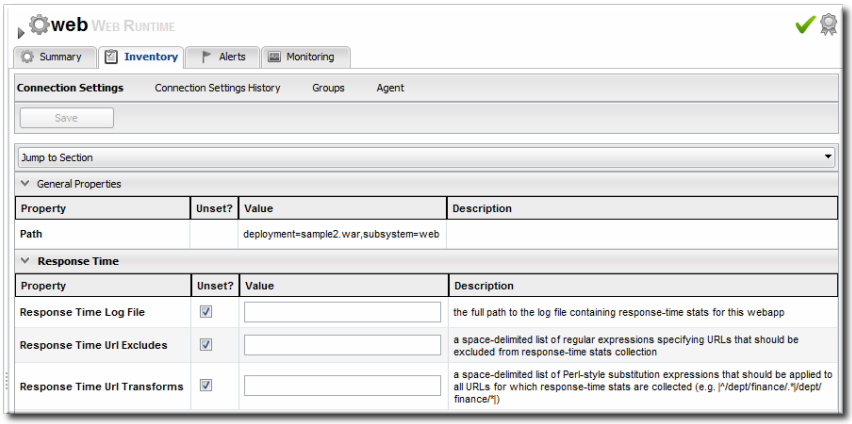


Figure 9.3. Web Runtime Resource

1. Click the **`Inventory`** tab in the top menu.
2. Click the **`Servers - Top Level Imports`** item, and select the JBoss EAP 6 resource.
3. Navigate to the deployment resource, and expand the application to the web subsystem.
4. Click the **`Monitoring`** tab on the web resource entry.
5. Click the **`Schedules`** subtab.
6. Select the **`Response Time`** metric. This metric is the *calltime* type.



7. Click the **`Enable`** at the bottom of the list.
8. Click the **`Inventory`** tab on the web entry.
9. Select the **`Connection Settings`** subtab.
10. Unset the check boxes for the response time configuration and fill in the appropriate values for the web application.



- The response times log which is used by that specific web application. **The log file is a required setting for call-time data collection to work.**.
- Any files, elements, or pages to exclude from response time measurements. The response times log records times for all resources the web server serves, including support files like CSS files and icons or background images.
- The same page can be accessed with different parameters passed along in the URL. The **`Response Time Url Transforms`** field provides a regular expression that can be used to strip or substitute the passed parameters.

9.3. Setting up Monitoring for EAP 6 Resources

The **`Runtime`** tab in the JBoss EAP 6 management console gives an current snapshot of the state of managed servers and some common subsystems. The emphasis of the EAP 6 console is on quick access to current information. This makes it easy to perform immediate management tasks.

JBoss ON focuses on long-term performance tracking, not just an immediate view. The EAP 6 agent plug-in for JBoss ON allows administrators to monitor additional metrics for resources, to track historical measurements and establish performance baselines, and to view how the resource status has changed (gone up or down) and how long it has been in different states.

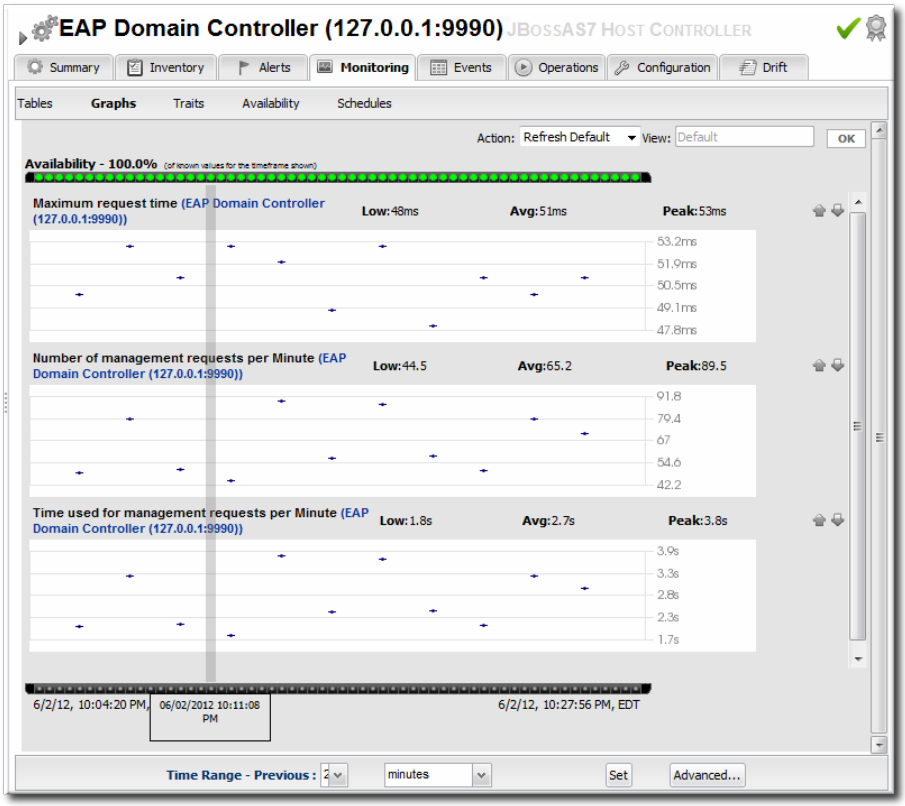


Figure 9.4. Monitoring Graph

Monitoring information is displayed in graphs and in tables on the **`Monitoring`** tab. The **`Summary`** tab has visually different line graphs rendered in the monitoring portlet.

NOTE

For more information on monitoring concepts and configuration, see "Setting up Monitoring, Alerts, and Operations".

9.3.1. Enabling Additional Metrics

JBoss ON supports all of the metrics for JBoss EAP 6 resources that are displayed in the EAP 6 console, along with many additional metrics.

The metrics enabled by default in JBoss ON, however, may not be the same ones that are viewable in the EAP 6 console. For example, for data sources, the EAP 6 console displays the available connections count, active connections count, and max used connections count. In JBoss ON, the default metrics collected are the maximum and minimum pool size settings, the max wait time, and the number of time outs per minute.

NOTE

A complete list of possible metrics for each resource type is given in the Resource Reference.

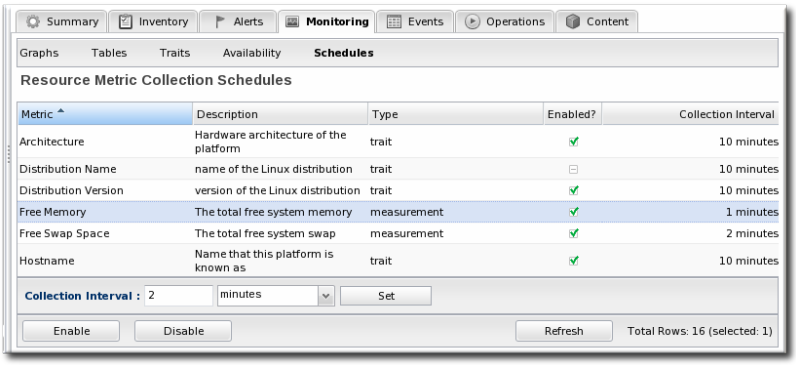
Metric collection can be enabled or disabled for a resource. Administrators can also reset how often JBoss ON checks a given metric (the *collection interval*).

TIP

Checking less important metrics less frequently can improve platform and agent performance, while still recording that information for reference or event correlation.

To change the monitoring configuration for a resource:

1. Click the **`Inventory`** tab in the top menu.
2. Select **`Servers - Top Level Imports`** in the **`Resources`** menu table on the left. Then select the JBoss EAP 6 server and navigate to the resource to edit.
3. Click the **`Monitoring`** tab on the resource entry.
4. Click the **`Schedules`** subtab.
5. To enable a metric:
    a. Select the metric row. Multiple metrics can be selected using the **`Ctrl`** key.
    b. Click the **`Enable`** button at the bottom of the page.
6. To change a collection interval for a metric:
    a. Select the metric row. Multiple metrics can be selected using the **`Ctrl`** key, if they will all be changed to the same frequency.



    b. Enter the desired collection period in the **`Collection Interval`** field, with the appropriate time unit (seconds, minutes, or hours).
    c. Click **`Set`**.

NOTE

Changing the collection interval for a disabled metric automatically enables the metric.

9.3.2. Availability Monitoring

Availability is a (mostly) straightforward measurement that shows whether a resource is running and minimally responsive. If the agent can connect to the resource, that resource is available.

This is how availability is displayed in the EAP 6 console for a resource: it shows whether a resource is currently running.

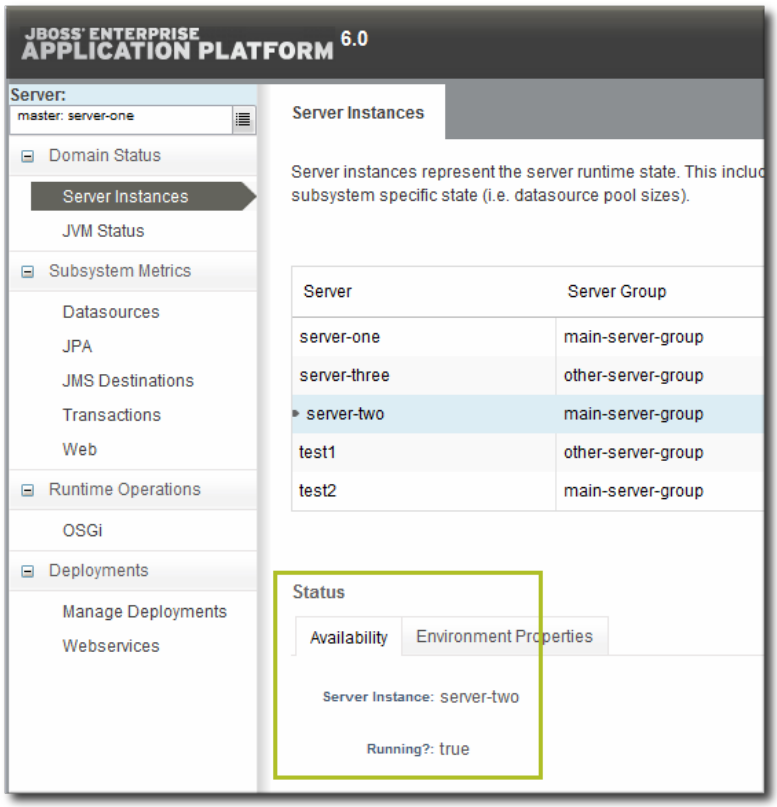


Figure 9.5. Availability in the EAP 6 Console

As a measure of performance, availability can acquire more shades of gray than simply *is it running*. For example, has a resource been cycling off and on repeatedly? Does it stay offline for long periods before it recovers? Is a resource unavailable because of its own failure or is a parent resource (like the platform) unreachable, while the child is running fine? JBoss ON supports four resources states — up, down, disabled, and unknown — to try to differentiate between real states for the resources.

Availability is covered in much more detail in the availability chapter of "Setting up Monitoring, Alerts, and Operations".

Along with showing the current status of a resource, JBoss ON also shows how long the resource has been in that state and what percentage of time the resource spends in different states. This allows JBoss ON to calculate a few different trends per resource:

- Its overall uptime percentage
- Total time spent in different states (up, down, or disabled)
- The total number of failures (times it has gone down)
- The mean time between failures, which is essentially the mean time that a resource is up and available
- The mean time to recovery, which is the mean amount of time that it takes a resource to come back up after a failure

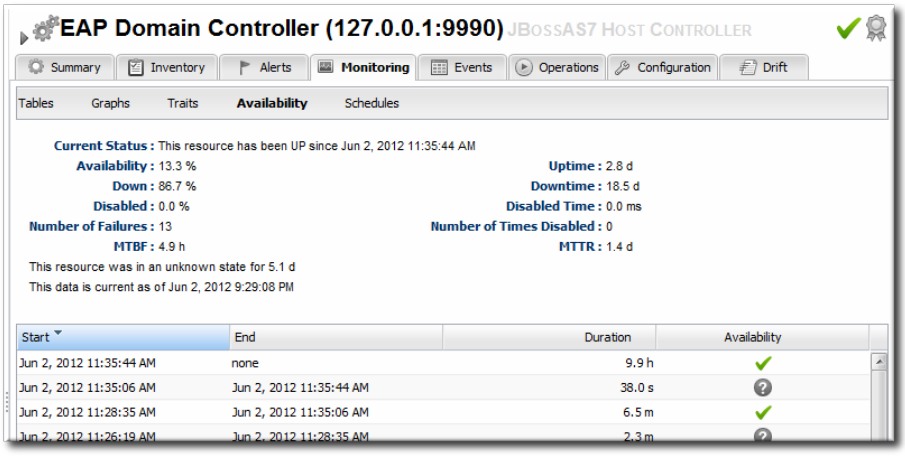


Figure 9.6. Availability in JBoss ON

Availability is collected on a schedule, much like a monitoring metric. Controlling the frequency of availability checks can establish a resource priority.

NOTE

Servers, such as domain controllers and managed servers, have availability checks that run every minute. This can help catch resource cycling in unstable servers. Services, like datasources and JMS queues, are checked for availability every 10 minutes.

The frequency of availability checks can have an impact on agent performance, particularly for servers structured like EAP 6, with a large number of child resources. Every resource is checked for availability, by default. There may be some low-level child resource, like OSGi classes or EJBs, that really do not require an independent availability check. In that case, availability checks can be disabled for the child. The agent automatically applies the parent state to the child; if the parent is running, than the child is assumed to be running. This is called *backfilling*.

9.4. Configuring Events Monitoring

Event monitoring is essentially a log filter. The resource itself maintains a log file; JBoss ON checks any specified log file for messages or events that occur. JBoss ON can check for messages in a stream or only for messages that match a certain pattern or severity.

JBoss ON can issue alerts based on log messages and even take action automatically in response to events, like creating a new managed resource or restarting a server.

Since events are unpredictable, based on the real operations of the resource rather than a schedule, it is a big advantage to administrators to be able to filter messages and respond dynamically.

NOTE

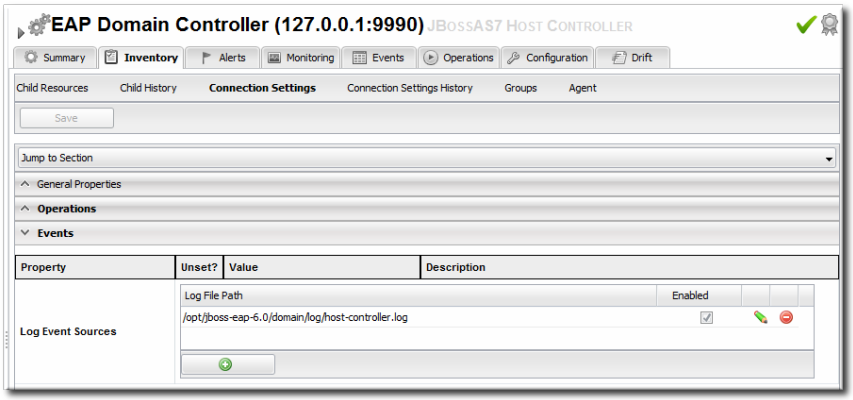
Events configuration is described more in the events chapter of "Setting up Monitoring, Alerts, and Operations".

Three JBoss EAP 6 server types support events, and they are preconfigured with an event log in the standard location:

- Host controllers, in **`domain/log/host-controller.log`**
- Managed servers, in **`domain/servers/server-three/log/server.log`**
- Standalone servers, in **`standalone/log/server.log`**

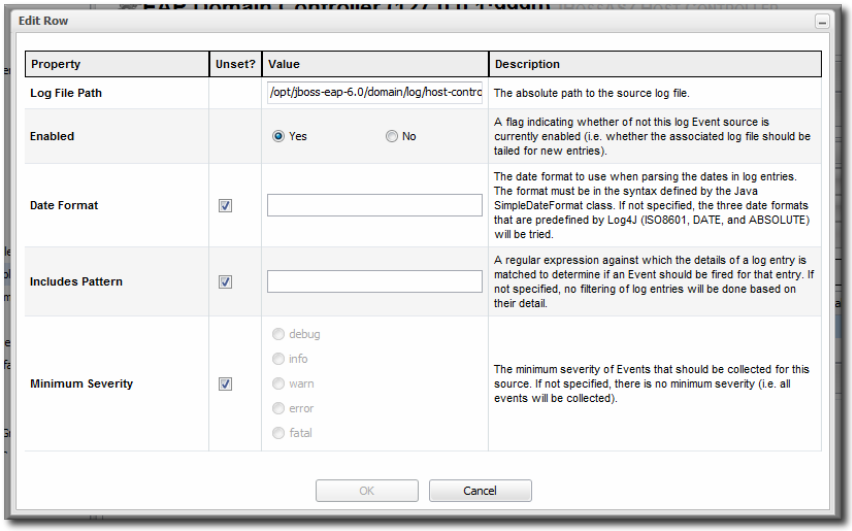
While the event log is already configured, it is not enabled by default. The event log must be enabled before events are collected by the JBoss ON agent. Additional settings, such as specific strings to look for in log messages, can also be set.

1. Click the **`Inventory`** tab in the top menu.
2. Click the **`Servers - Top Level Imports`** item, and select the JBoss EAP 6 resource.
3. Navigate to the appropriate EAP 6 resource.
4. Click the **`Inventory`** tab on the resource entry.
5. Select the **`Connection Settings`** subtab.
6. Click green plus icon under the **`Events Log`** section.



7. Enable the event entry.

   Optionally, set the date format, string filters to include messages, or the severity of the messages to include.


Edit Row
Property
Unset?
Value
Description
Log File Path
/opt/jboss-eap-6.0/domain/log/host-contro
The absolute path to the source log file.
Enabled
Yes
No
A flag indicating whether of not this log Event source is currently enabled (i.e. whether the associated log file should be tailed for new entries).
Date Format
The date format to use when parsing the dates in log entries. The format must be in the syntax defined by the Java SimpleDateFormat class. If not specified, the three date formats that are predefined by Log4J (ISO8601, DATE, and ABSOLUTE) will be tried.
Includes Pattern
A regular expression against which the details of a log entry is matched to determine if an Event should be fired for that entry. If not specified, no filtering of log entries will be done based on their detail.
Minimum Severity
debug
info
warn
error
fatal
The minimum severity of Events that should be collected for this source. If not specified, there is no minimum severity (i.e. all events will be collected).
OK
Cancel

When event monitoring is enabled, the matching log messages are displayed in the **Events** tab for the resource.

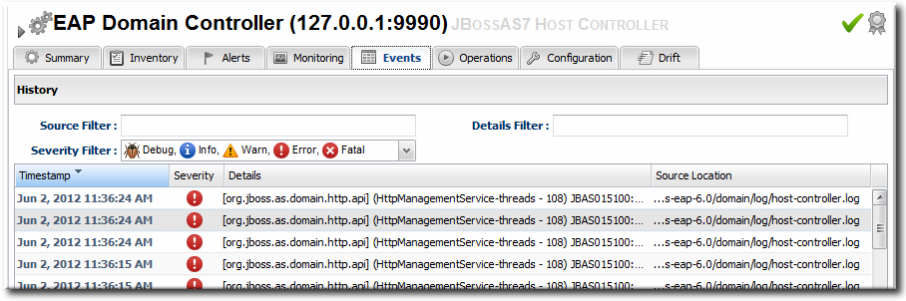


Figure 9.7. JBoss EAP 6 Event Logging

9.5. Alerting on JBoss EAP 6 Resources

Alerting is covered in detail in the alerting and notifications chapters of "Setting up Monitoring, Alerts, and Operations", and that is the best reference for details on planning what alert conditions to set and how to respond.

The key thing in planning alerts is planning automatic responses to whatever condition triggered the alert. For example, within an EAP 6 domain, one managed server may go offline or begin exhibiting extremely slow response times. In that case, JBoss ON can automatically create a new managed server instance, within the same server group, on another platform to take the failing server's place.

Taking an immediate response, without necessarily requiring administrator intervention, is invaluable for customer-facing problems, such as poor response times for a web application.

There are several types of responses that JBoss ON can take:

- Email an administrator
- Run a restart operation on the alerting resource
- Run an operation on the parent of the alerting resource
- Run a shell script
- Run a JBoss ON server-side script

  Server-side scripts are extremely powerful. A script can perform almost any management task in JBoss ON: change resource configuration, deploy updated packages, create new resources, delete existing resources, run an operation, or all of the above. For more information on writing server-side scripts, see "Writing JBoss ON Command-Line Scripts".

Chapter 10. Using the mod_cluster Services in EAP 6

The **mod_cluster** modules provides dynamic load balancing between web contexts on a JBoss application server and an Apache web server.

There are two halves to creating a cluster with **mod_cluster**: configuring the module on JBoss to manage the web applications and configuring the module on Apache to manage sessions and routing.

JBoss ON can manage the embedded **mod_cluster** subsystem for JBoss EAP 6 for both domain servers and standalone servers.

10.1. About mod_cluster and JBoss ON

The **mod_cluster** module, a subsystem on EAP 6, communicates between the web applications on a JBoss EAP server and an Apache web server. Multiple JBoss EAP servers can be involved in a **mod_cluster** group, and those servers can be managed servers, standalone servers, or a mix of both.

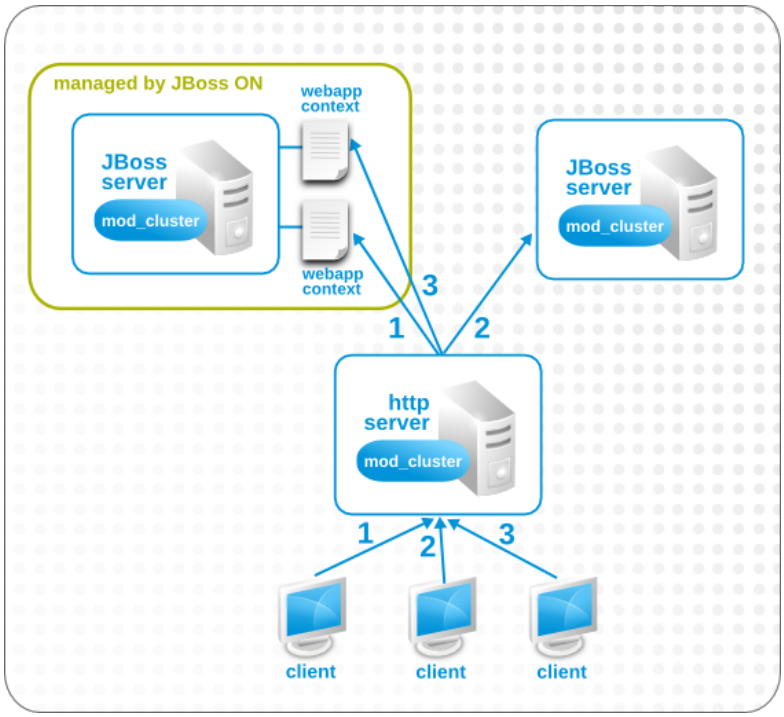


Figure 10.1. The mod_cluster Topology

Only high availability profiles contain the **mod_cluster** module. For domains, the **other-server-group** server group is configured to use the **full-ha** profiles, though any profile which supports high availability can be used (such as the **ha** profile or a custom profile). For standalone servers, the server must be started with the **standalone-ha.xml** configuration.

One EAP server within the **mod_cluster** server is the master node; that is the administering **mod_cluster** service. Every other member of the cluster is a worker node.

NOTE

Information about **mod_cluster** in general is available with the mod_cluster project documentation.

Whether a specific resource belongs to the **mod_cluster** domain depends on the profile that resource is associated with, which is largely outside of the control of JBoss ON. (Of course, a standalone server can be started with the high availability configuration with the **JAVA_OPTS** settings in JBoss ON for start scripts, or a new managed server can be created that uses a high availability profile. But the profile definitions themselves are created and maintained outside of JBoss ON.)

JBoss ON manages the **mod_cluster** configuration itself. The cluster settings include multicast (advertising), load balancing, session handling, and network settings.

10.2. Configuring Multicast for Load Balancing

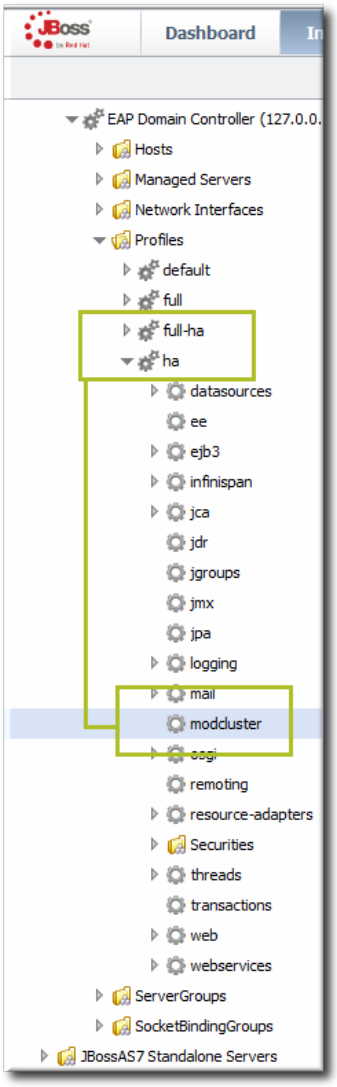
mod_cluster can use multicast signals to inform nodes which proxy servers are available. This allows nodes to register themselves automatically to the domain.

As a subset of the registration process, nodes can be directed to a specific domain or load balancer.

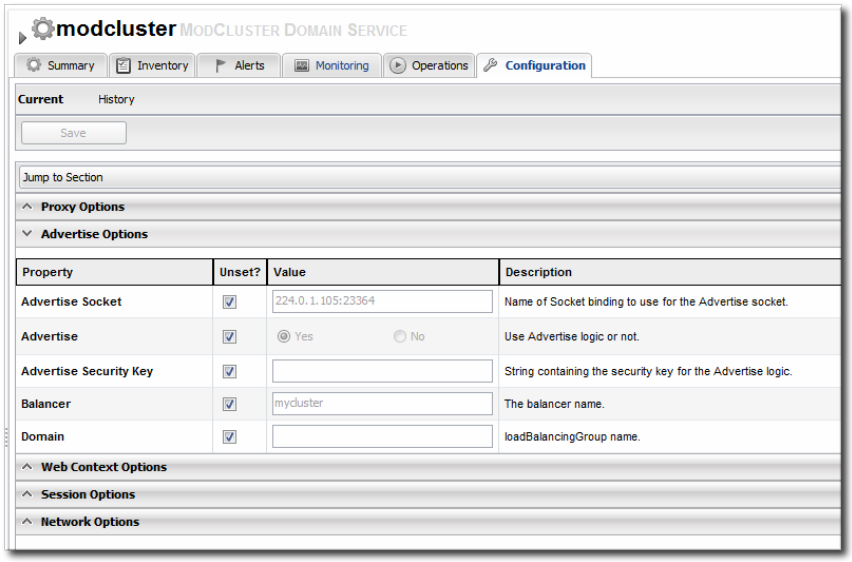
NOTE

Multicast is configured by default.

1. Click the **Inventory** tab in the top menu.
2. Select **Servers - Top Level Imports** in the **Resources** menu table on the left. Then select the JBoss EAP 6 server which hosts the **mod_cluster** service (the master node).
3. Navigate to the **mod_cluster** service resource.



4. Click the **Configuration** tab on the resource entry.
5. Go to the **Advertise Options** section.



6. Change the multicast settings.

   To use a specific load balancer for the **`mod_cluster`** nodes, set the **`Balancer`** field to the load balancer name and, optionally, the **`Domain`** field to the load balancer group, which is one of the groups configured on the balancer itself.

   IMPORTANT

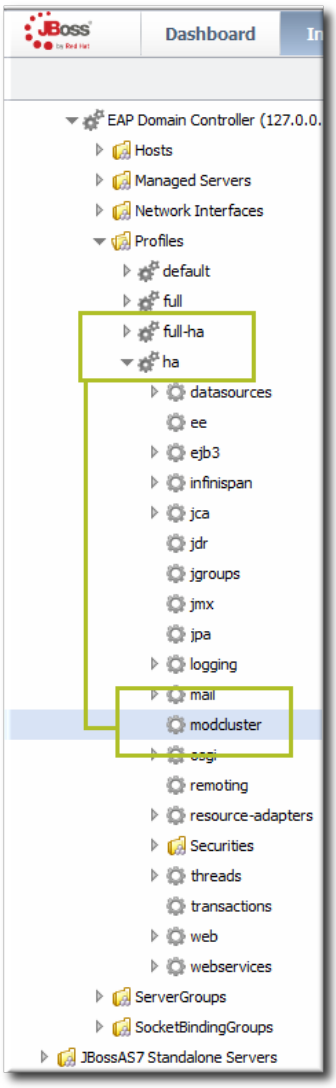
   The **`Balancer`** and **`Domain`** values must match the configuration for the corresponding Apache server.

7. Click the **`Save`** button at the top of the page.

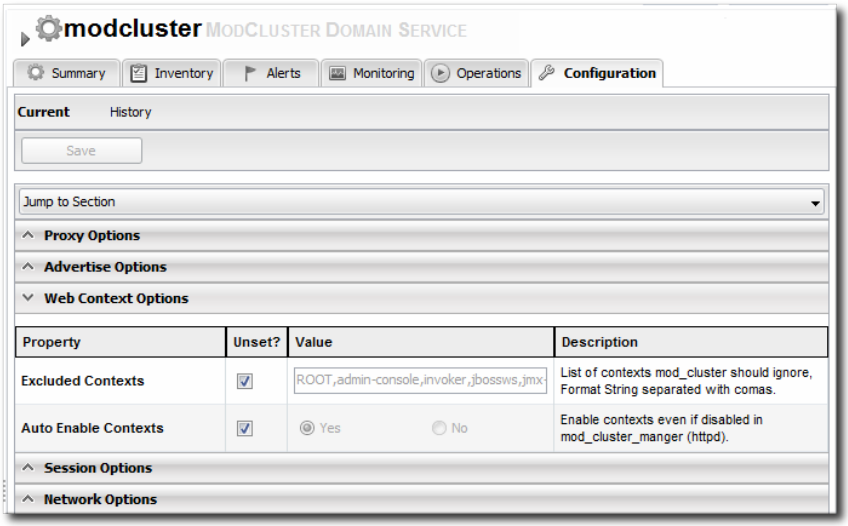
10.3. Excluding Web Contexts from Discovery

By default, any web context on a node is discovered and managed by the **`mod_cluster`** service. If a web context should not be included in the cluster, it can be excluded, by name.

1. Click the **`Inventory`** tab in the top menu.
2. Select **`Servers - Top Level Imports`** in the **`Resources`** menu table on the left. Then select the JBoss EAP 6 server which hosts the **`mod_cluster`** service (the master node).
3. Navigate to the **`mod_cluster`** service resource.



4. Click the **`Configuration`** tab on the resource entry.
5. Go to the **`Web Context Options`** section.



6. Unset the **`Excluded Contexts`** field, and add the names of any contexts to exclude.

NOTE

Some internal contexts for JBoss EAP are excluded by default. This should be kept in the excludes list; any new contexts to exclude should be added to the existing list.

7. Click the **`Save`** button at the top of the page.

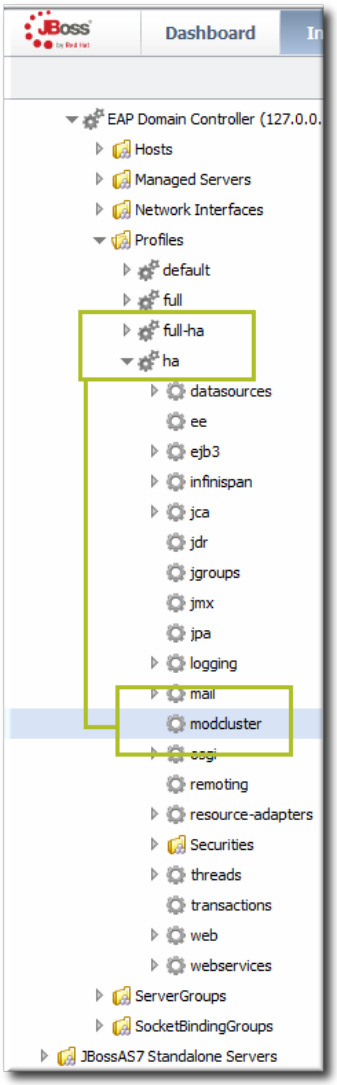
10.4. Configuring Web Context Metrics

mod_cluster can use different types of metrics to determine how to balance traffic most efficiently. These metrics are described in the mod_cluster documentation. These metrics include things like active sessions, request counts, and traffic.

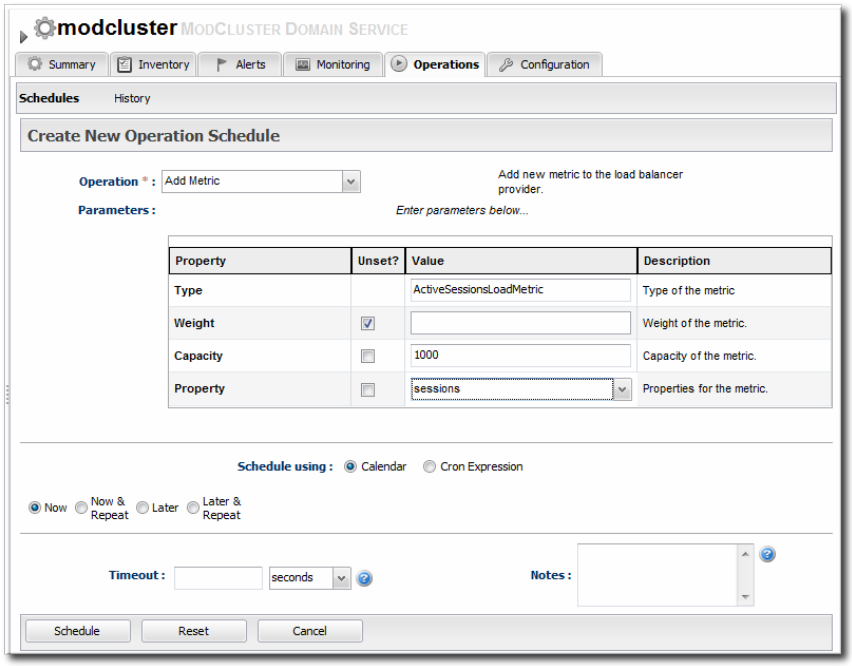
Additionally, custom metrics can be written and added to the subsystem.

These metrics can be added to the **mod_cluster** subsystem configuration for JBoss ON to monitor (and existing metrics can be removed).

1. Click the **Inventory** tab in the top menu.
2. Select **Servers - Top Level Imports** in the **Resources** menu table on the left. Then select the JBoss EAP 6 server which hosts the **mod_cluster** service (the master node).
3. Navigate to the **mod_cluster** service resource.



4. Click the **Operations** tab on the resource entry.
5. Select the **Add (Custom) Metrics** operation from the drop-down menu.



6. Fill in the metric information. The default metrics are listed in the mod_cluster documentation.
7. Click the **`Schedule`** button at the bottom of the page.

Part III. Managing JBoss EAP 5

Chapter 11. Discovering JBoss EAP/AS 5 Servers

Most JBoss EAP/AS servers are configured in a way that allows them and their child resources to be discovered and monitored without any additional configuration. When the agent attempts to discover and then manage resources, it uses certain assumptions based on common configuration to identify and connect to those resources. If for some reason the JBoss EAP/AS server has configuration that differs from those assumptions, then JBoss ON may not be able to manage that system fully.

This section describes configuration settings that are required to discover or manage certain JBoss EAP/AS server resource types.

11.1. Discovering and Managing the JBoss AS/EAP 5 JVM

For the JBoss ON agent to discover the JVM resource and its subsystems (such as memory, threading, and logging) for a JBoss EAP resource, the JBoss EAP instance must be configured to use the platform MBean server as the location where it registers its MBeans.

If necessary, edit the JBoss EAP/AS server start script so that it uses the platform MBean server.

1. Open the **`run.conf`** file. For example, on Red Hat Enterprise Linux:

   ```
   [root@server ~]# vim jbossInstallDir/bin/run.conf
   ```

2. Add the **`JAVA_OPTS`** settings to use the platform MBean server.

   On Red Hat Enterprise Linux:

   ```
   JAVA_OPTS="$JAVA_OPTS -Djboss.platform.mbeanserver"
   ```

   On Windows:

   ```
   set JAVA_OPTS=%JAVA_OPTS% -Djboss.platform.mbeanserver
   ```

3. Restart the JBoss EAP/AS server.

11.2. Enabling Remote Access to JMX and Profile Service

Management tasks like monitoring and running operations require that the agent is able to connect to the JMX server for the JBoss EAP/AS instance. This means that remote JMX access must be enabled.

1. Verify that the JBoss Naming Protocol server (JNP) is deployed. (It is deployed by default.)

   Open the **`jboss-service.xml`** file:

   ```
   [root@server ~]# vim jbossInstallDir/server/config/conf/jboss-service.xml
   ```

   Then, make sure that there is a line enabling the JNP connector.

   ```
   <mbean code="org.jboss.naming.NamingService"> ... </mbean>
   ```

2. While not required, it is recommended that you enable authentication for the JNP service. (This is enabled by default.)

   a. Open the **`jmx-invoker-service.xml`** file.

   ```
   [root@server ~]# vim jbossInstallDir/server/config/deploy/jmx-invoker-
   service.xml
   ```

   b. Add a line for the JMX connector authentication.

   ```
   <interceptor
   code="org.jboss.jmx.connector.invoker.AuthenticationInterceptor"
                       securityDomain="java:/jaas/jmx-console"/>
   ```

   c. Make sure that the admin user is listed in the JMX users properties file. When a new JBoss EAP resource is discovered, the agent reads the JMX username and password from this file and stores them in the discovered JBoss EAP resource's connection settings. These settings are then used to connect to the EAP server's JNP service.

   ```
   [root@server ~]# vim jbossInstallDir/server/config/conf/props/jmx-
   console-users.properties
   ```

   d. Uncomment or add the user information. This is a simple key/value pair, *username=password*. For example:

   ```
   admin=admin
   ```

3. Restart the JBoss EAP/AS server.

11.3. Setting Start Script Arguments, Environment Variables, and JAVA_OPTS

11.3.1. Start Script Discovery and Settings

As part of discovering a JBoss EAP 5 server, JBoss ON attempts to discover the environment that the server is running in. Specifically, the discovery process attempts to discover the runtime environment:

- The discovery process identifies, or attempts to identify, the start script that was used to start the EAP server, whether it was **`run.sh|bat`** or a custom start script.
- Discovery detects a subset of environment variables set in the **`run.conf`** file or parent process that are required for the start script to work.

NOTE

Although the discovery process does detect some environment variables, **the discovery scan does not `JAVA_OPTS` values.**
The connection properties for the start script intentionally defer to the **`run.conf`** file for **`JAVA_OPTS`** values.

- Discovery attempts to detect any arguments passed to the start script itself.

The discovered settings are stored in the **`Start Script Environment Variables`** and **`Start Script Arguments`** connection settings for the EAP 5 resource.

If the discovery scan cannot detect some of the script settings (such as the agent is running as a different user than the EAP 5 server and cannot read the parent process), it fails gracefully. It simply gathers whatever information it can and ignores blank values.

NOTE

The **`Start Script Environment Variables`** and **`Start Script Arguments`** connection settings are only discovered once, when the resource is initially discovered. After that, the connection settings can be changed in the connection settings configuration, but any changes made on the local system will not be detected. For example, if the server is discovered with a particular value (like **`-XX:PermSize=256M`**), the argument value will not be updated if the server is restarted later with a different setting value.

The script arguments and environment variables are the only ones used by the agent when it runs the start script. These arguments and environment variables are not added to other configuration settings or the parent process. The start script settings in the EAP 5 connection settings are deterministic.

In previous versions of the plug-in, the EAP/AS 5 server had a Java home directory setting and bind address (hostname or IP) that was used by the agent as part of identifying and running the start scripts.

If the **`Script Arguments`** field is empty, then the bind address and Java home directory [2] are still used for invoking the start script. If *any* script arguments are set, then the bind address and Java home values are ignored.

The script arguments value overrides any other start script connection settings.

11.3.2. Start Script Arguments and Drift Monitoring

JBoss ON can monitor directories or specific files to check for *configuration drift*, which means configuration changes that move away from a designated configuration state. This is described in more detail in the drift chapters of "Managing Resource Configuration."

While drift monitoring is critical for administrators to manage important resources, there may be times when it is necessary or beneficial from that configuration in a few settings. The start script arguments can be used to override the defined configuration for an EAP 5 server *without* triggering a drift alert.

Since the start script options are all connection settings, every change is recorded in a change history and can be easily viewed and reverted. This keeps the system properties and Java settings trackable and remediable.

Example 11.1. System Properties Without Violating the Drift Definition

Tim the IT Guy creates one specific set of configuration that all production EAP 5 servers should use an environment. He then creates a drift definition template for monitoring and associates or pins that blessed configuration to the template.

Every EAP 5 server which uses that drift template must conform to those configuration settings. Tim the IT Guy creates an alert that informs him if any production server drifts away from that specified configuration. This is an important safety measure for Example Co.'s production application servers.

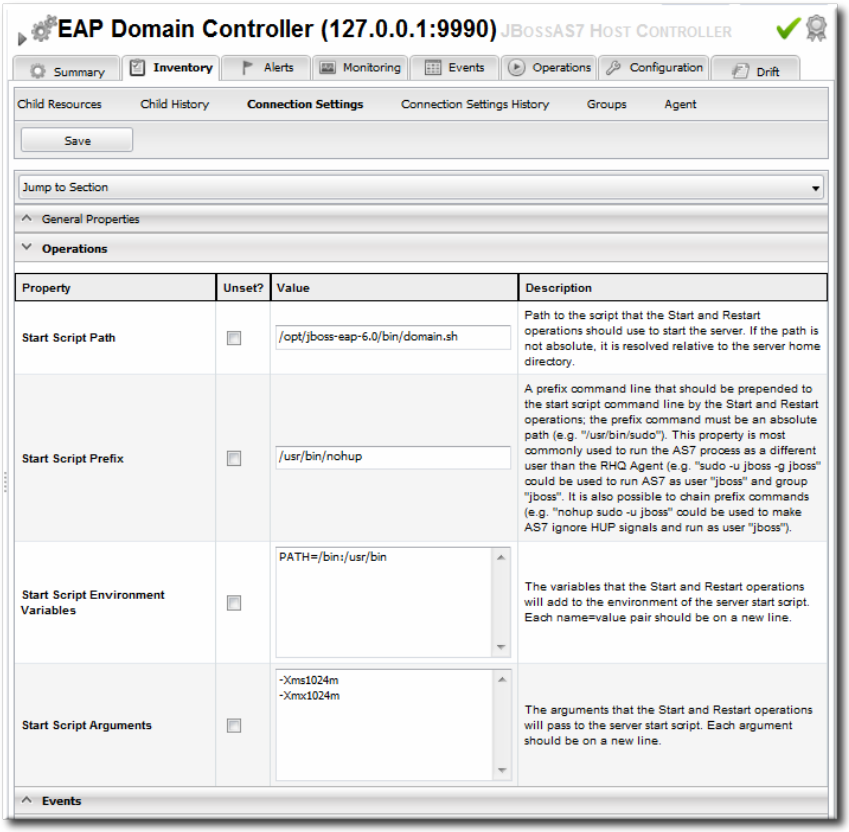
However, a couple of the production servers have slightly different hardware and other applications running on them, so they require different heap sizes to run effectively.

If Tim the TI Guy adds a system property to use a different heap size, he is going to receive constant drift alerts or his edited configuration could be overwritten if he runs an automatic server-side script to remediate drift configuration.

By setting different heap settings through the start script, Tim can apply the right settings for that system without editing a configuration file, so there is no alert-able configuration drift.

11.3.3. Changing Start Script Configuration

1. Click the **Inventory** tab in the top menu.
2. Select **Servers - Top Level Imports** in the **Resources** menu table on the left. Select the JBoss EAP 5 server.
3. In the inventory tree, select the top resource entry for the server.
4. Open the **Inventory** tab, and select the **Connection Settings** subtab.
5. Expand the **Operations** section.
6. Change or add start script settings. These are the scripts and settings that the JBoss ON agent uses when running a start or restart operation on the EAP 5 server.


EAP Domain Controller (127.0.0.1:9990)
JBossAS7 Host Controller
Summary
Inventory
Alerts
Monitoring
Events
Operations
Configuration
Drift
Child Resources
Child History
Connection Settings
Connection Settings History
Groups
Agent
Save
Jump to Section
General Properties
Operations
Property
Unset?
Value
Description
Start Script Path
/opt/jboss-eap-6.0/bin/domain.sh
Path to the script that the Start and Restart operations should use to start the server. If the path is not absolute, it is resolved relative to the server home directory.
Start Script Prefix
/usr/bin/nohup
A prefix command line that should be prepended to the start script command line by the Start and Restart operations; the prefix command must be an absolute path (e.g. "/usr/bin/sudo"). This property is most commonly used to run the AS7 process as a different user than the RHQ Agent (e.g. "sudo -u jboss -g jboss" could be used to run AS7 as user "jboss" and group "jboss". It is also possible to chain prefix commands (e.g. "nohup sudo -u jboss" could be used to make AS7 ignore HUP signals and run as user "jboss").
Start Script Environment Variables
PATH=/bin:/usr/bin
The variables that the Start and Restart operations will add to the environment of the server start script. Each name=value pair should be on a new line.
Start Script Arguments
-Xms1024m
-Xmx1024m
The arguments that the Start and Restart operations will pass to the server start script. Each argument should be on a new line.
Events

   - To use a custom start script, enter the path and script name.
   - Optionally, enter a prefix to use with the script when running the start script.
   - Set any environment variables, one per line.
   - Set any script arguments, one per line. For regular JAVA_OPTS, these arguments usually are **-X**, **-D**, or **-P**. Some useful **-XX** arguments are listed in the JVM options documentation from Sun.

     The EAP 5 default start scripts use a **run.sh**-style script, so the arguments use that format. A custom script can use different arguments or options.
7. Click the **Save** button at the top of the page.

[2] The JBoss EAP/AS 5 server's Java home setting overrides the agent Java setting, if that start script value is set.

Chapter 12. Creating JBoss EAP 5 Child Resources

12.1. Creating Data Sources

NOTE

It can take several minutes for the new child resource to be added and visible in the JBoss ON inventory because the new resource has to be created on the agent platform and then discovered by the agent.

1. Search for the JBoss server instance to which to deploy the data source.
2. On the details page for the selected JBoss server instance, open the **Inventory** tab.
3. In the **Create New** drop-down menu, select the item for **- Data Sources**.
4. Select a template for the data source. There are three data sources templates to populate a data source with common information:
   - The default template is used with SQL databases like PostgreSQL or MySQL.
   - The Oracle Local TX is used for Oracle databases with local transactions.
   - The Oracle XA template is used for Oracle databases with XA transactions.
5. Along with the obvious settings, like the resource name, enter the information for the specific child resource to be deployed:
   - The type of data source to create, either No TX Data Sources, Local TX Data Sources or XA Data Sources
   - A unique JNDI name for the DataSource wrapper to use to bind under
   - The fully qualified name of the JDBC driver or DataSource class, such as **org.postgresql.Driver**
   - The JDBC driver connection URL string, such as **jdbc:postgresql://127.0.0.1:5432/foo**
   - The username and password to use to connect to the data source
   - The minimum and maximum connection pool sizes for this data source

   Additional settings are available under the **Advanced Settings** area.

12.2. Creating Connection Factories

NOTE

It can take several minutes for the new child resource to be added and visible in the JBoss ON inventory because the new resource has to be created on the agent platform and then discovered by the agent.

1. Search for the JBoss server instance to which to deploy the connection factory.
2. On the details page for the selected JBoss server instance, open the **Inventory** tab.
3. In the **Create New** drop-down menu, select the item for **- Connection Factory**.
4. Along with the obvious settings, like the resource name, enter the information for the specific child resource to be deployed:
   - The type of connection factory to create, either tx-connection-factory (transaction) or no-tx-connection-factory (no transaction)
   - A unique JNDI name for the DataSource wrapper to use to bind under
   - The username and password to use to connect to the data source
   - The minimum and maximum connection pool sizes for this data source

   Additional settings are available under the **Advanced Settings** area.

12.3. Creating JMS Queues and Topics

JMS Queues and JMS Topics are child resources of a JBossMQ service or JBossMessaging service resource, which itself is a child of a JBoss server instance.

NOTE

It can take several minutes for the new child resource to be added and visible in the JBoss ON inventory because the new resource has to be created on the agent platform and then discovered by the agent.

1. Search for the JBoss messaging service to which to deploy the JMS queue or topic.
2. On the details page for the selected JBoss messaging service, open the **Inventory** tab.
3. In the **Create New** drop-down menu, select the **- JMQ JMS Topic** or **- JMQ JMS Queue** item.
4. Aside from the obvious settings, like the resource name, the JMS Queue or JMS Topic entry

requires two additional parameters:

- The name of the queue or topic to use as the JMX object name
- A unique JNDI name for the DataSource wrapper to use to bind under

Additional settings are available under the **`Advanced Settings`** area.

Chapter 13. Deploying Applications

Applications that are deployed on a JBoss server are a special type of resource. They have a place in the JBoss ON inventory and are children of the JBoss AS/EAP server. In that way, they can be discovered or created as a resource. However, some JBoss child resources — EARs and WARs — are also content, so they are treated as software packages that are stored and then deployed to JBoss servers. For these applications, the child resource is created first, by uploading a package to the JBoss server. After that, they are managed like content, with updated packages added to a content repo and then applied to the JBoss server.

13.1. Setting Permissions for Agent and Resource Users

The assumption is that the JBoss ON agent and resources like a JBoss server or Tomcat server run as the same system user. This allows the agent and the application server itself to manage resource content and configuration simultaneously.

However, if the agent user is different than the resource user, then there can be problems when one entity makes a configuration change and the other attempts a change later.

For example, when deploying an application, the deployment operation is initiated by the agent and the content is supplied through the agent, and then the application server completes the actual deployment. When deleting an application, the application server handles the undeployment by itself.

If a WAR file is deployed exploded without a **`MANIFEST.MF`** file, the agent creates one when it writes the SHA-256 value for the package. When the JBoss EAP/AS server server tries to remove the WAR application later (and the JBoss EAP/AS server user is different than the agent user), then the removal fails. The JBoss EAP/AS server server cannot delete the **`MANIFEST.MF`** file. The agent then rediscovers the application directory and re-initiates the deployment operation for the removed WAR.

NOTE

This situation only occurs when the application is exploded *and* does not contain the **`MANIFEST.MF`** file — meaning, a situation where the agent creates a file within the deployment directory. Even if the agent and JBoss EAP/AS server users are different, this situation does not occur if the application is not exploded or where the agent does not write any files.

This situation can be avoided. If the agent user and resource user are different, then change the system settings:

1. Add the agent user and the resource user to the same primary group.
2. Set the **`umask`** value for the agent user to give read and write permissions to both user and group, such as **`660`**. For example:

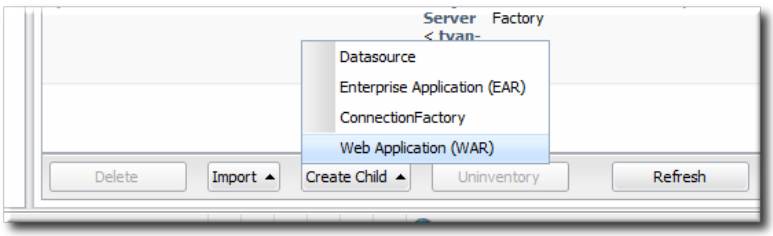
```
vim /home/rhqagent/.bashrc

umask 660
```

13.2. Deploying EAR and WAR Files

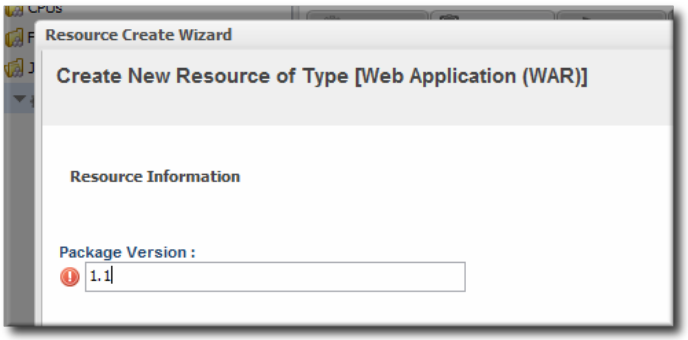
IMPORTANT

Newly-deployed content may not show up in the JBoss ON inventory for as long as 24 hours, even if it was successfully created. By default, discovery scans for services are only made every 24 hours.
To see it immediately, run an *execute prompt command* operation on the agent and enter the **`discovery`** command. This runs a discovery scan.

1. Search for the JBoss server instance to which to deploy the EAR or WAR.
2. On the details page for the selected JBoss server instance, open the **`Inventory`** tab.
3. In the **`Create New`** menu at the bottom, select the item for **- Web Application (WAR)** or **- Enterprise Application (EAR)**, as appropriate.



4. Enter the version number.



This is not used for the resource. The actual version number is calculated based on the spec version and implementation version in **MANIFEST.MF**, if any are given, or the calculated SHA-256 value for the package itself:

```
SPEC(IMPLEMENTATION)[sha256=abcd1234]
```

If no version numbers are defined in **MANIFEST.MF**, then the SHA value is used. The SHA value is always used to identify the package version internally.

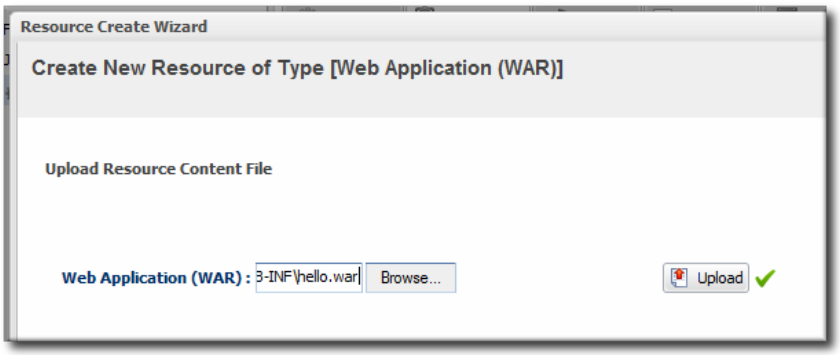
NOTE

When the EAR or WAR file is exploded after it is deployed, the **MANIFEST.MF** file is updated to include the calculated SHA version number. For example:

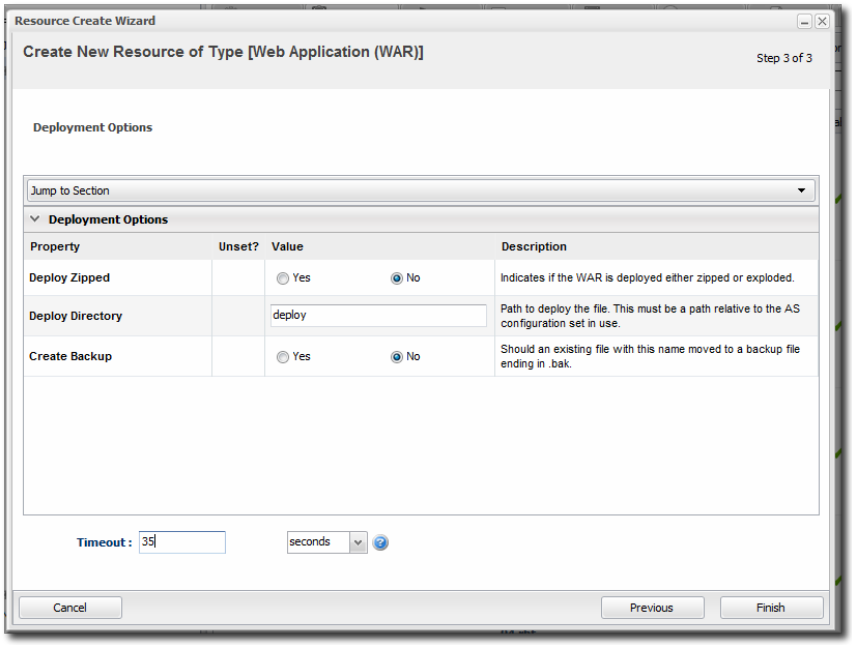
```
Manifest-Version: 1.0
Created-By: Apache Maven
RHQ-Sha256: 570f196c4a1025a717269d16d11d6f37
...
```

For more information on package versioning, see "Deploying Applications and Content".

5. Upload the EAR/WAR file.



6. Enter the information for the application to be deployed.



- Whether the file should be exploded (unzipped) when it is deployed.
- The path to the directory to which to deploy the EAR or WAR package. The destination directory is relative to the JBoss server instance installation directory; this cannot contain an absolute path or go up a parent directory.
- Whether to back up any existing file with the same name in the target directory.

Once the EAR/WAR file is confirmed, the new child resource is listed in the **`Child History`** subtab of the **`Inventory`** tab.

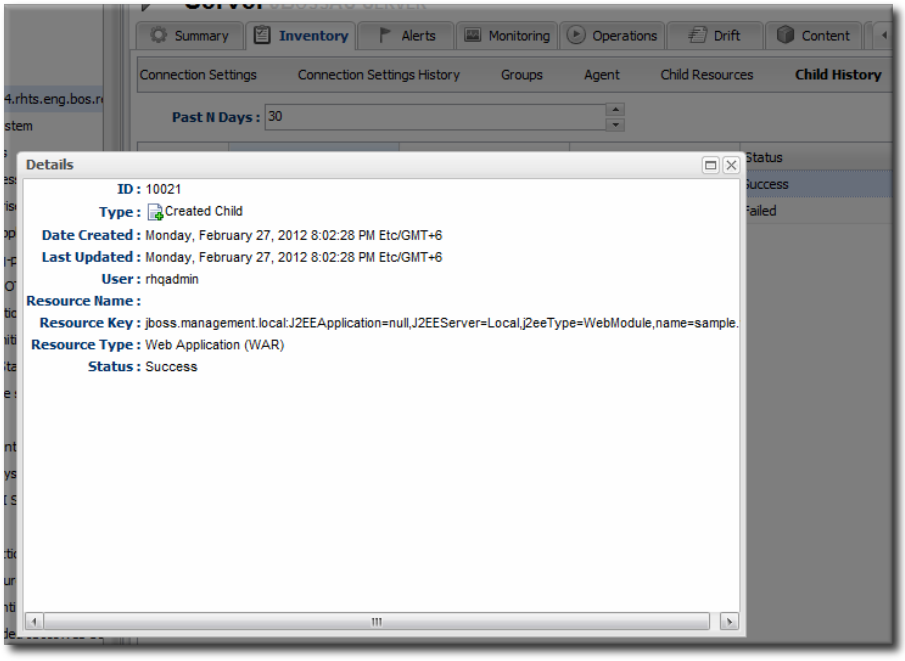
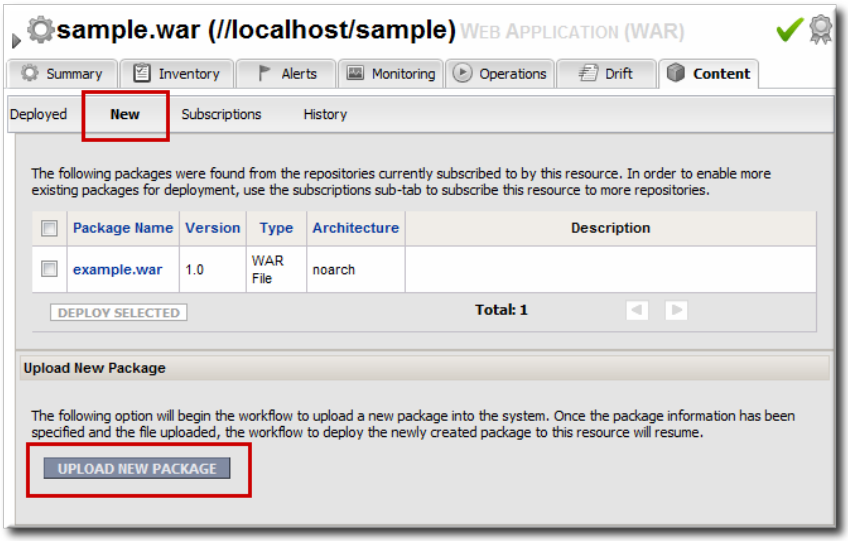


Figure 13.1. WAR Child Resource

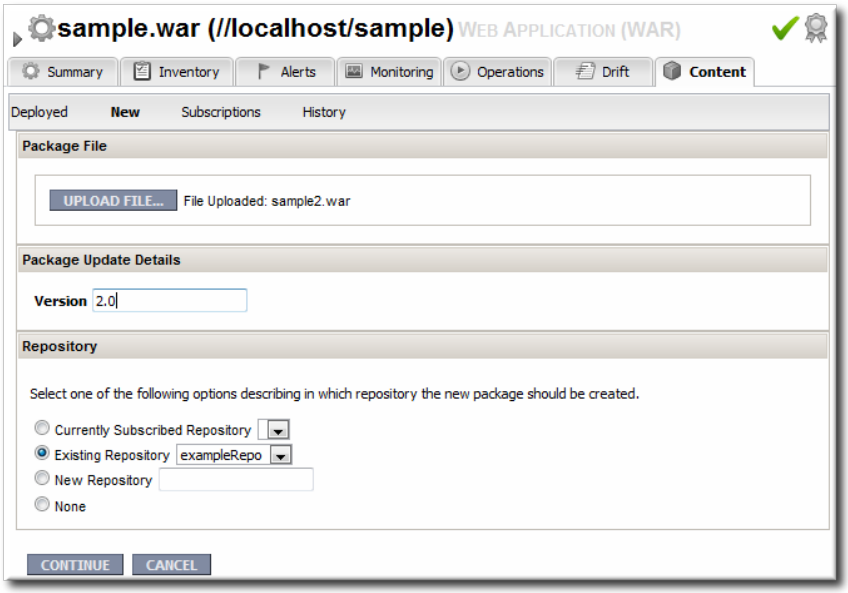
13.3. Updating Applications

After the EAR or WAR resource is created, changes are treated like updated content packages. Updating the EAR/WAR resource is the same as uploading and applying new packages to that EAR/WAR resource entry.

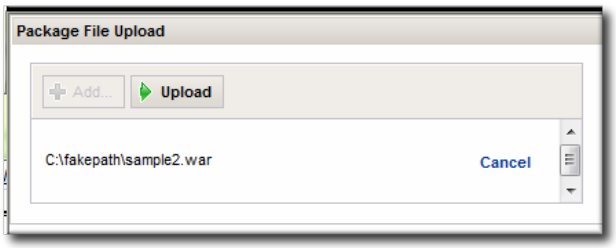
1. Browse to the EAR or WAR resource in the JBoss ON UI.
2. In the EAR or WAR resource details page, open the **`Content`** tab, and click the **`New`** subtab.


sample.war (//localhost/sample) WEB APPLICATION (WAR)
Summary
Inventory
Alerts
Monitoring
Operations
Drift
Content
Deployed
New
Subscriptions
History
The following packages were found from the repositories currently subscribed to by this resource. In order to enable more existing packages for deployment, use the subscriptions sub-tab to subscribe this resource to more repositories.
Package Name
Version
Type
Architecture
Description
example.war
1.0
WAR File
noarch
DEPLOY SELECTED
Total: 1
Upload New Package
The following option will begin the workflow to upload a new package into the system. Once the package information has been specified and the file uploaded, the workflow to deploy the newly created package to this resource will resume.
UPLOAD NEW PACKAGE

3. Click the **`UPLOAD NEW PACKAGE`** button.
4. Click the **`UPLOAD FILE`** button.



5. In the pop-up window, click the **`Add`** button, and browse the local filesystem to the updated WAR or EAR file to be uploaded.

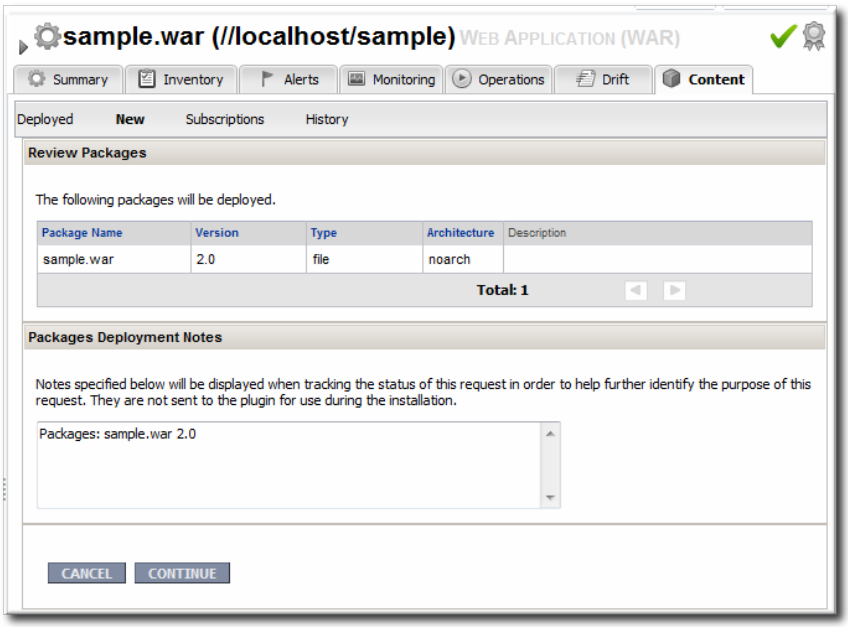


6. Click the **`UPLOAD`** button to load the file and dismiss the window.
7. In the main form, select the repository where the WAR or EAR file package should be stored. If one exists, select an existing repository or a subscribed repository for the resource. Otherwise, create a new repository.
8. Optionally, set the version number for the EAR/WAR package.

   If this is set, then this value is displayed in the UI. If not, then a version number is calculated, based on the spec version and implementation version in **`MANIFEST.MF`**, if any are given, or the calculated SHA-256 value for the package itself. Internally, the package is identified by the SHA value.

   ```
   SPEC(IMPLEMENTATION)[sha256=abcd1234]
   ```

   For more information on package versioning, see "Deploying Applications and Content".
9. Confirm the details for the new package, then click **`CONTINUE`**.



When the package is successfully uploaded, the UI redirects to the history page on the **`Content`** tab.

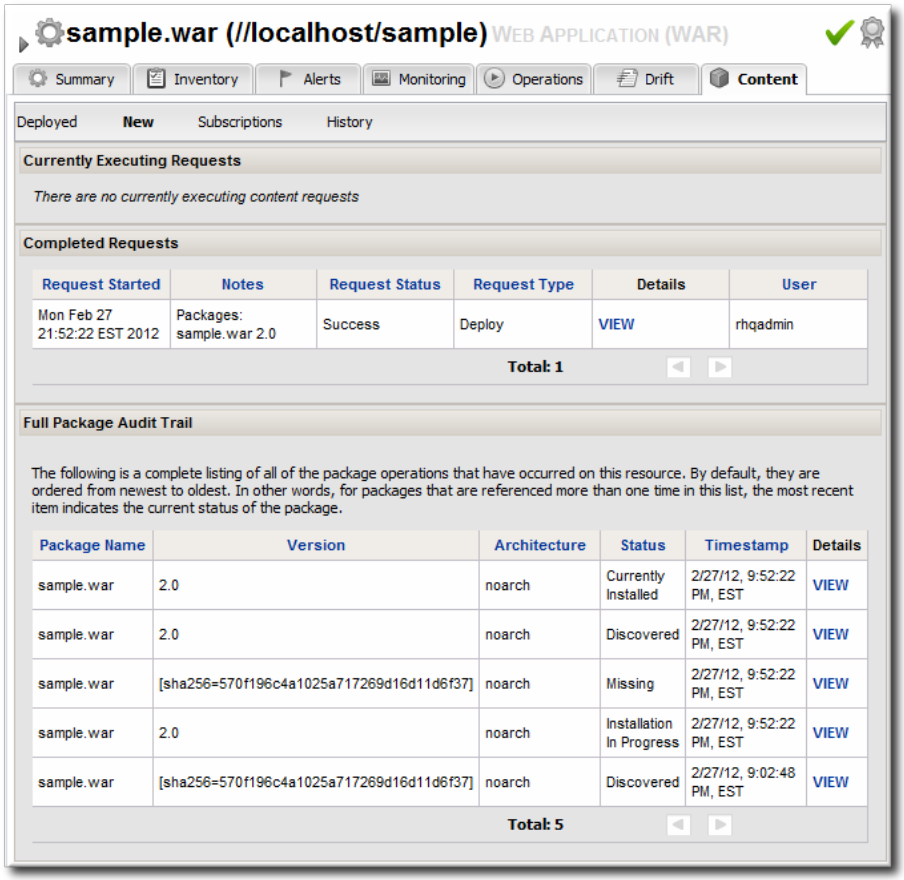
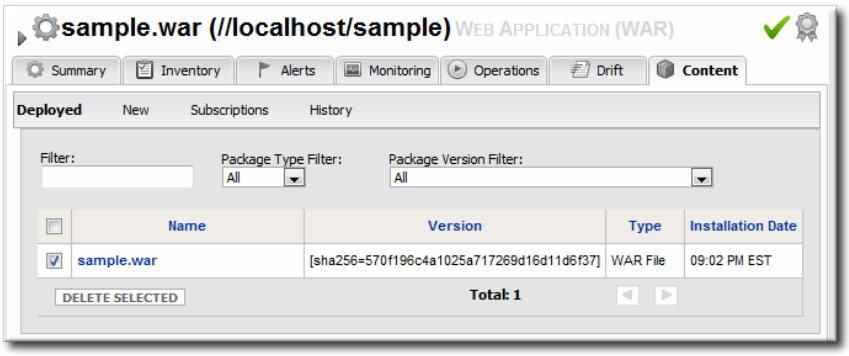


Figure 13.2. Deployment History for a Resource

13.4. Deleting an Application

Deleting an EAR/WAR application is the same as deleting the currently deployed package associated with that EAR/WAR resource entry.

1. Browse to the EAR or WAR resource in the JBoss ON UI.
2. In the EAR or WAR resource details page, open the **`Content`** tab, and click the **`Deployed`** subtab.
3. Select the checkbox by the EAR/WAR package, and click the **`DELETE SELECTED`** button.


sample.war (//localhost/sample) WEB APPLICATION (WAR)
Summary
Inventory
Alerts
Monitoring
Operations
Drift
Content
Deployed
New
Subscriptions
History
Filter:
Package Type Filter:
All
Package Version Filter:
All
Name
Version
Type
Installation Date
sample.war
[sha256=570f196c4a1025a717269d16d11d6f37]
WAR File
09:02 PM EST
DELETE SELECTED
Total: 1

Chapter 14. Applying JBoss Patches from the Patch RSS Feed

Content management can be used to apply cumulative patches to JBoss AS, JBoss EAP, JBoss SOA-P, and JBoss EWP instances. By default, the JBoss ON server comes pre-configured with the JBoss Patch Content Source and JBoss Patch Repository.

14.1. Planning How to Patch JBoss Resources

JBoss ON can automatically apply patches as they are available in the JBoss Customer Portal (CP). When a patch is released, an RSS feed on the Customer Portal is updated, and JBoss ON monitors that feed and lists the latest patches in the GUI.

There are two parts to planning how to apply patches:

- Identifying and configuring content sources
- Planning how to execute manual steps

14.1.1. Identifying Content Sources

A *content source* connects a JBoss ON server to some kind of content delivery mechanism. With JBoss products, the content source is the Customer Portal, which contains all patches for all JBoss products. (A content source can be another content repository, depending on your products and environment.)

A content source identifies a location. A *repository* within JBoss ON maps a content source to the resources in the inventory that the patches are applied to.

For JBoss patches, the default content provider connects the JBoss ON server to the cumulative patches provided by the JBoss Customer Service Portal. The default repository associated with the content provider is where the metadata about the patches and the patches themselves are stored within JBoss ON.

The JBoss ON agent is the entity which actually executes the patching process on a resource. The agent is informed of updates, pulls the information from the server, and then updates the local JAR and class files within the managed JBoss instance. The patching process runs independently of any server configuration profile or base configuration.

JBoss products can receive and apply patch updates through the JBoss Customer Portal feed in JBoss ON. The supported products include:

- JBoss Enterprise Application Platform (EAP)
- JBoss Enterprise Web Platform and Web Services (EWP and EWS)
- JBoss Enterprise Data Services (EDS)
- JBoss SOA Platform (SOA-P)

NOTE

A Customer Portal feed is only available for a product or a specific version of a product if there is a patch in the Customer Portal for that product. JBoss ON depends on the JBoss Customer Portal to provide patch information.

14.1.2. Planning Manual Steps

Some patches require additional, manual changes, such as editing an XML configuration file. There are several different situations that require manual intervention:

- The file to be patched is not present in the configuration.
- There are files that need to be removed manually.
- Configuration files, such as XML or Java properties files, require patches that need to be applied manually.
- Seam is being used and must be patched manually.

Basically, admin intervention is required to resolve anything that is outside the default configuration, like merging in custom configuration or updating custom libraries.

JBoss ON performs the standard steps required to apply patches to a JBoss instance, but it does not (and should not) have any way to parse and then merge changes in the configuration. JBoss ON does not attempt to determine, value, and apply custom changes. That sort of heuristic is best performed by an administrator.

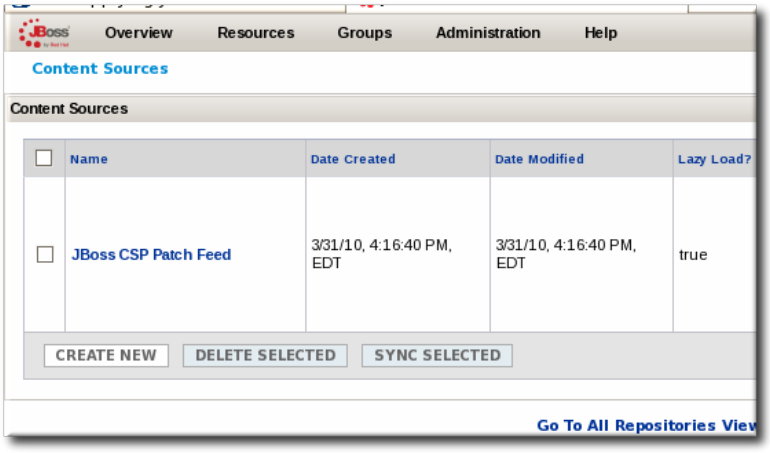
Any manual steps which are required to complete the patch are listed in the content update summary after the patch is applied.

14.2. Enabling the Default JBoss Patch Content Source

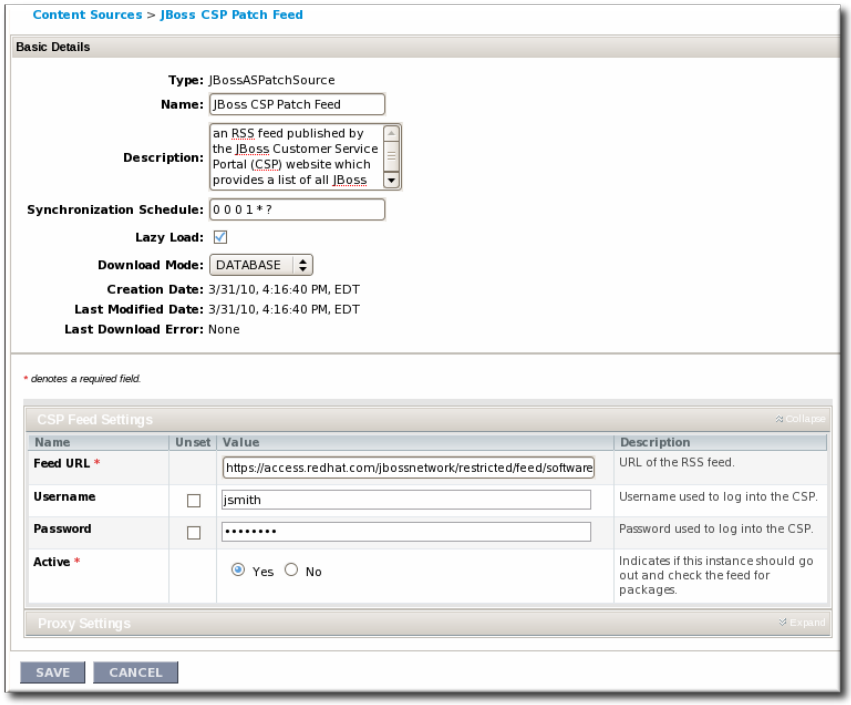
TIP

Perform patch installations during off hours or scheduled maintenance periods.

1. In the **Administration** tab in the top menu, open the **Content** menu and select the **Content Sources** item.
2. Click the **JBoss Customer Portal Patch Feed** source.



3. Click the **Edit** button at the bottom of the **Customer Portal Feed Settings** area to modify the content source.
4. Fill in the required connection information.



   - The Customer Portal username and password.

NOTE

The Customer Portal password is obfuscated when it is stored in the JBoss ON database.

   - The URL for the content source (https://access.redhat.com/jbossnetwork/restricted/feed/software.html?product=all&downloadType=all&flavor=rss&version=&jonVersion=2.0).
   - The activation state. This should be **Yes** to enable automatic patching.

Most of the default settings, such as the schedule, can be kept.

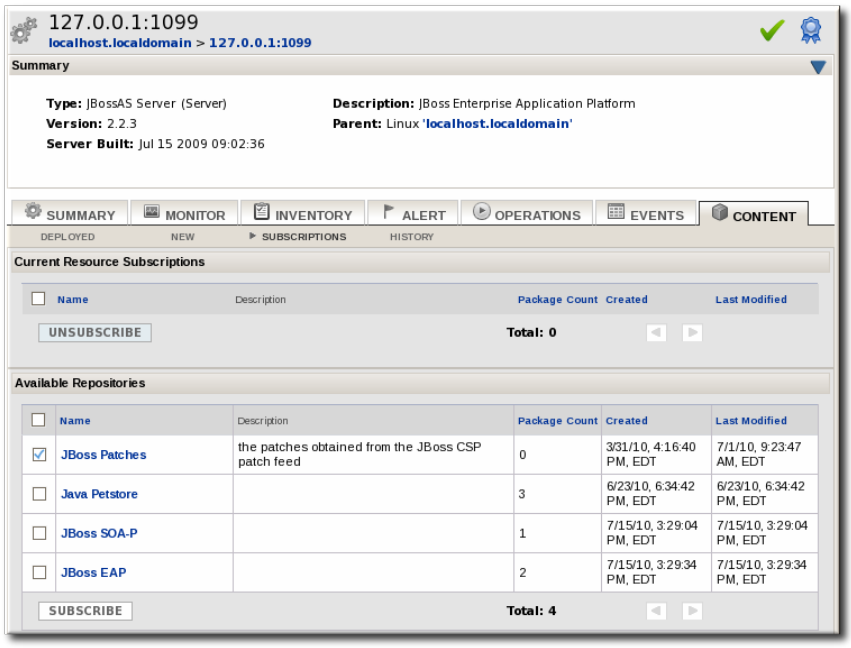
IMPORTANT

Keep the **Lazy Load** checkbox activated, or all patches defined in the RSS feed, 1 GB of data, is preemptively downloaded by the JBoss ON server.

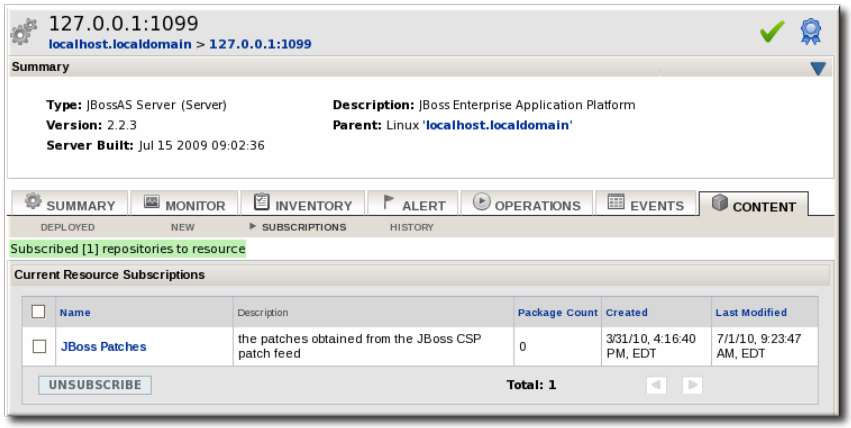
5. Click **Save**.
6. Optionally, use **Synchronize** button to force the content source to pull down the latest RSS Feed and synchronize it with the local data. The history of synchronization attempts is listed in the **Synchronization Results section**.
7. Complete any manual steps (Section 14.1.2, "Planning Manual Steps") to finish applying the patches.

14.3. Subscribing a Specific Resource to the Default JBoss Patch Repository

1. In the **Resources** item in the top menu, go to the **Servers** item.
2. Search for the JBoss instance to subscribe to the repository.
3. On the JBoss server resource's entry page, open the **Content** tab and select the **Subscriptions** subtab. The JBoss patches repository is listed as available for subscription.


127.0.0.1:1099
localhost.localdomain > 127.0.0.1:1099
Summary
Type: JBossAS Server (Server)
Version: 2.2.3
Server Built: Jul 15 2009 09:02:36
Description: JBoss Enterprise Application Platform
Parent: Linux 'localhost.localdomain'
SUMMARY MONITOR INVENTORY ALERT OPERATIONS EVENTS CONTENT
DEPLOYED NEW SUBSCRIPTIONS HISTORY
Current Resource Subscriptions
Name Description Package Count Created Last Modified
UNSUBSCRIBE
Total: 0
Available Repositories
Name Description Package Count Created Last Modified
JBoss Patches the patches obtained from the JBoss CSP patch feed 0 3/31/10, 4:16:40 PM, EDT 7/1/10, 9:23:47 AM, EDT
Java Petstore 3 6/23/10, 6:34:42 PM, EDT 6/23/10, 6:34:42 PM, EDT
JBoss SOA-P 1 7/15/10, 3:29:04 PM, EDT 7/15/10, 3:29:04 PM, EDT
JBoss EAP 2 7/15/10, 3:29:34 PM, EDT 7/15/10, 3:29:34 PM, EDT
SUBSCRIBE
Total: 4

4. Select the checkbox beside the JBoss patches repository, and click the **SUBSCRIBE** button. When the assignment is complete, the repository is listed under the **Current Resource Subscriptions** area, rather than **Available Repositories**.


127.0.0.1:1099
localhost.localdomain > 127.0.0.1:1099
Summary
Type: JBossAS Server (Server)
Version: 2.2.3
Server Built: Jul 15 2009 09:02:36
Description: JBoss Enterprise Application Platform
Parent: Linux 'localhost.localdomain'
SUMMARY MONITOR INVENTORY ALERT OPERATIONS EVENTS CONTENT
DEPLOYED NEW SUBSCRIPTIONS HISTORY
Subscribed [1] repositories to resource
Current Resource Subscriptions
Name Description Package Count Created Last Modified
JBoss Patches the patches obtained from the JBoss CSP patch feed 0 3/31/10, 4:16:40 PM, EDT 7/1/10, 9:23:47 AM, EDT
UNSUBSCRIBE
Total: 1

14.4. Subscribing Multiple JBoss Resources to the Default JBoss Patch Repository

The repository is associated with a content source (the patches that can be applied) and resources are then subscribed to this repository (where the patches can be applied to).

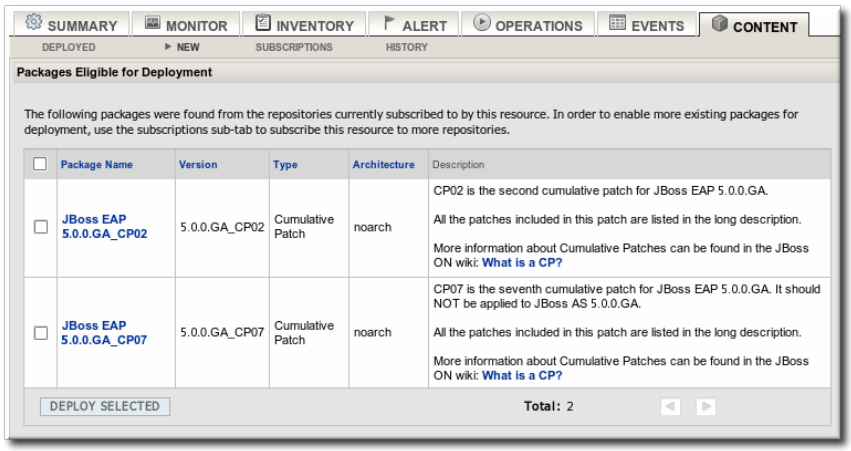
1. In the **Administration** tab in the top menu, open the **Content** menu and select the **Repositories** item.
2. Click the **JBoss Patch Channel**.

   The JBoss patch repository is associated with the JBoss patch content source by default. To associate the repository to another content source, click the **ASSOCIATE...** button and assign the content source.
3. In the main page for the repository, click the **SUBSCRIBE...** button to subscribe JBoss resources to the patch repository.
4. In the search area, select **Server** in the drop-down menu.
5. Select all the JBoss server instance resources to subscribe to this repository.

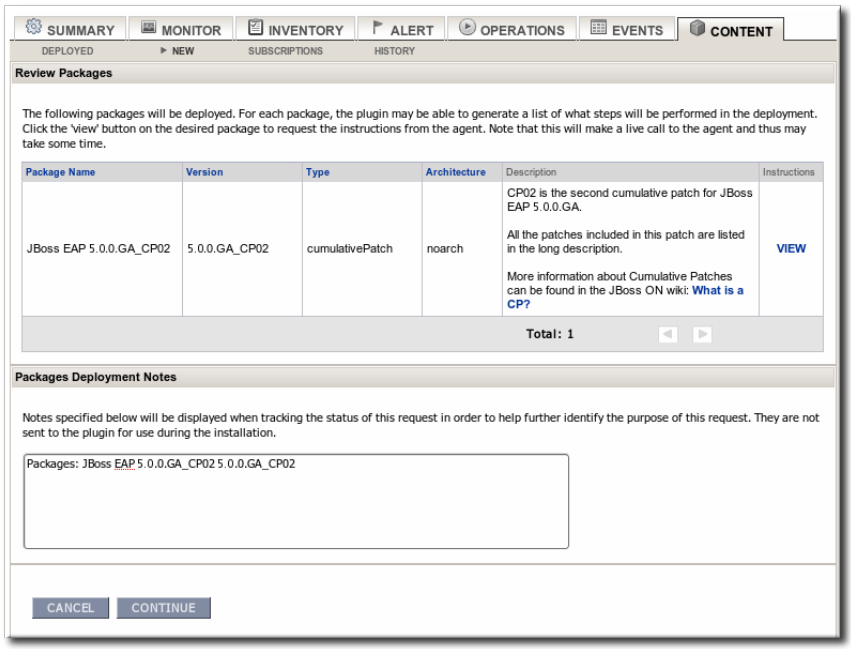
14.5. Applying a Patch

For patches to be applied to a JBoss server, the server must first be subscribed to the JBoss content repository.

1. Stop the JBoss instance.
2. In the **Resources** item in the top menu, go to the **Servers** item.
3. Search for the JBoss instance to patch.
4. On the JBoss server resource's entry page, open the **Content** tab and select the **New** subtab. This lists all of the packages and patches which are available for that specific resource type.
5. Select the checkboxes beside the names of the patched to install, and click the **DEPLOY SELECTED** button.

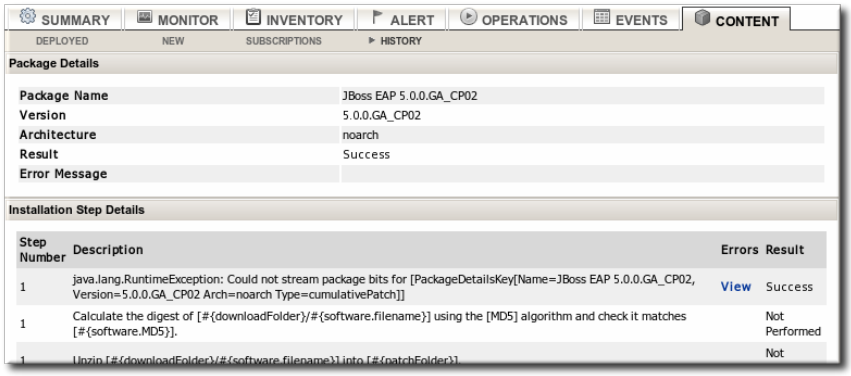

SUMMARY
MONITOR
INVENTORY
ALERT
OPERATIONS
EVENTS
CONTENT
DEPLOYED
NEW
SUBSCRIPTIONS
HISTORY
Packages Eligible for Deployment
The following packages were found from the repositories currently subscribed to by this resource. In order to enable more existing packages for deployment, use the subscriptions sub-tab to subscribe this resource to more repositories.
Package Name
Version
Type
Architecture
Description
JBoss EAP 5.0.0.GA_CP02
5.0.0.GA_CP02
Cumulative Patch
noarch
CP02 is the second cumulative patch for JBoss EAP 5.0.0.GA.
All the patches included in this patch are listed in the long description.
More information about Cumulative Patches can be found in the JBoss ON wiki: What is a CP?
JBoss EAP 5.0.0.GA_CP07
5.0.0.GA_CP07
Cumulative Patch
noarch
CP07 is the seventh cumulative patch for JBoss EAP 5.0.0.GA. It should NOT be applied to JBoss AS 5.0.0.GA.
All the patches included in this patch are listed in the long description.
More information about Cumulative Patches can be found in the JBoss ON wiki: What is a CP?
DEPLOY SELECTED
Total: 2

6. Review the information on the page and verify everything is correct. Click the **VIEW** link in the **Instructions** column to review the steps that will be performed during the package installation process.


SUMMARY
MONITOR
INVENTORY
ALERT
OPERATIONS
EVENTS
CONTENT
DEPLOYED
NEW
SUBSCRIPTIONS
HISTORY
Review Packages
The following packages will be deployed. For each package, the plugin may be able to generate a list of what steps will be performed in the deployment. Click the 'view' button on the desired package to request the instructions from the agent. Note that this will make a live call to the agent and thus may take some time.
Package Name
Version
Type
Architecture
Description
Instructions
JBoss EAP 5.0.0.GA_CP02
5.0.0.GA_CP02
cumulativePatch
noarch
CP02 is the second cumulative patch for JBoss EAP 5.0.0.GA.
All the patches included in this patch are listed in the long description.
More information about Cumulative Patches can be found in the JBoss ON wiki: What is a CP?
VIEW
Total: 1
Packages Deployment Notes
Notes specified below will be displayed when tracking the status of this request in order to help further identify the purpose of this request. They are not sent to the plugin for use during the installation.
Packages: JBoss EAP 5.0.0.GA_CP02 5.0.0.GA_CP02
CANCEL
CONTINUE

7. Optionally, enter any notes to describe the patch deployment or environment.
8. Click the **CONTINUE** button to install the updates. The patch process runs in the background; the progress can be viewed in the **History** subtab of the **Content** tab.

9. When the installation process is complete, the patch job is listed in the **`Completed Requests`** area. Clicking the **`VIEW`** button displays the list of steps that performed in the process and whether they succeeded, failed, or were skipped.



10. When the patch process is complete, start the JBoss instance.

Chapter 15. Configuring HTTP Response Time Metrics for JBoss EAP/AS 5

15.1. Installing the Response Time Filters

To collect response time metrics for a JBoss EAP/AS 5 server, first install the JAR containing the JBoss ON response time servlet filter, and then configure the EAP/AS server to use it.

1. Download the Response Time servlet filter connector from the JBoss ON UI.

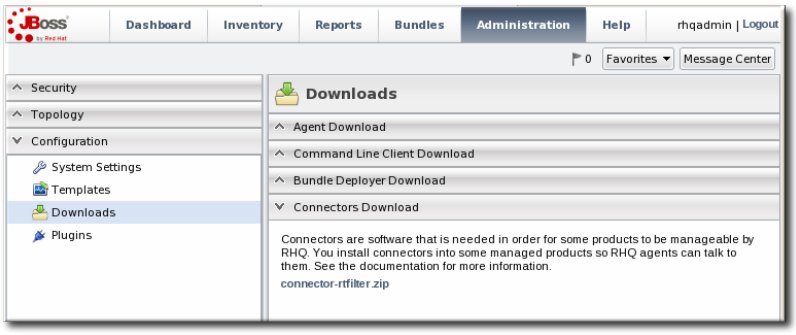
   TIP

   This can also be done from the command line using **wget**:

   ```
   [root@server ~]# wget
   http://server.example.com:7080/downloads/connectors/connector-
   rtfilter.zip
   ```

   a. Click the **Administration** tab in the top menu.

   b. In the **Configuration** menu box on the left, select the **Downloads** item.

   c. Click the **connector-rtfilter.zip** link, and save the file.

2. Unzip the connectors.

   ```
   [root@server ~]# unzip connector-rtfilter.zip
   ```

3. Copy the **rhq-rtfilter-***version***.jar** file into the **lib/** directory for the profile.

   ```
   [root@server ~]# cp connector-rtfilter/rhq-rtfilter-version.jar
   JbossHomeDir/lib/server/profileName/lib/
   ```

   JBoss EAP/AS already includes the **commons-logging.jar** file, which is also required for response time filtering.

4. Then configure the **web.xml** file for the EAP/AS instance.

   The response time filter can be deployed globally, for all web applications hosted by the EAP/AS instance or it can be configured for a specific web application.

   To configure it globally, edit the global **web.xml** file:

   ```
   [root@server ~]# vim
   JbossHomeDir/server/configName/default/deploy/jbossweb.sar/
   ```

   To configure it for a single web app, edit that one web app's **web.xml** file:

   ```
   [root@server ~]# vim WARLocation/WEB-INF/web.xml
   ```

5. Add the filter and, depending on the configuration, filter mapping elements to the file. This activates the response time filtering.

   All that is required for response time filtering to work is the default **<filter>** element, without any optional parameters. However, the parameters can be uncommented and set as necessary; the different ones are described in Table 15.1, “Parameters Available for User-Defined <filter> Settings”.

   TIP

   Instead of setting an optional parameters in the configuration file, they can be set in the configuration settings for the resource through the JBoss ON GUI.

```
<filter>
    <filter-name>RhqRtFilter</filter-name>
    <filter-class>org.rhq.helpers.rtfilter.filter.RtFilter</filter-class>

    <!-- Optional parameters.

    <init-param>
        <param-name>chopQueryString</param-name>
        <param-value>true</param-value>
    </init-param>
    <init-param>
        <param-name>logDirectory</param-name>
       <param-value>/tmp</param-value>
    </init-param>
    <init-param>
        <param-name>logFilePrefix</param-name>
        <param-value>localhost_7080_</param-value>
    </init-param>
    <init-param>
        <param-name>dontLogRegEx</param-name>
        <param-value></param-value>
    </init-param>
    <init-param>
       <param-name>matchOnUriOnly</param-name>
       <param-value>true</param-value>
    </init-param>
    <init-param>
        <param-name>timeBetweenFlushesInSec</param-name>
        <param-value>73</param-value>
    </init-param>
    <init-param>
        <param-name>flushAfterLines</param-name>
        <param-value>13</param-value>
    </init-param>
    <init-param>
        <param-name>maxLogFileSize</param-name>
        <param-value>5242880</param-value>
    </init-param>
    -->
</filter>

<!-- Use this only when also enabling the RhqRtFilter in the filter
<filter-mapping>
    <filter-name>RhqRtFilter</filter-name>
    <url-pattern>/*</url-pattern>
</filter-mapping>
-->
```

6. Restart the JBoss EAP/AS server to load the new **`web.xml`** settings.

Table 15.1. Parameters Available for User-Defined <filter> Settings

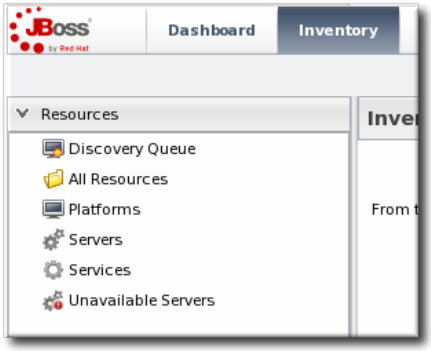
Parameter	Description
chopQueryString	Only the URI part of a query will be logged if this parameter is set to true. Otherwise the whole query line will be logged. Default is true.
logDirectory	The directory where the log files will be written to. Default setting is {**jboss.server.log.dir}/rt/** (usually **server/xxx/log/rt**). If this property is not defined, the fallback is {**java.io.tmpdir}/rt/** (**/tmp/** on UNIX®, and **~/Application Data/Local Settings/Temp** – check the **TEMP** environment variable) is used. If you specify this init parameter, no directory **rt/** will be created, but the directory you have provided will be taken literally.
logFilePrefix	A prefix that is put in front of the log file names. Default is the empty string.
dontLogRegEx	A regular expression that is applied to query strings. See java.util.regex.Pattern. If the parameter is not given or an empty string, no pattern is applied.
matchOnUriOnly	Should the dontLogRegEx be applied to the URI part of the query (true) or to the whole query string (false). Default is true.
timeBetweenFlushesInSec	Log lines are buffered by default. When the given number of seconds have passed and a new request is received, the buffered lines will be flushed to disk even if the number of lines to flush after (see next point) is not yet reached.. Default value is 60 seconds (1 Minute).
flushAfterLines	Log lines are buffered by default. When the given number of lines have been buffered, they are flushed to disk. Default value is 10 lines.
maxLogFileSize	The maximum allowed size, in bytes, of the log files; if a log file exceeds this limit, the filter will truncate it; the default value is 5242880 (5 MB).

vHostMappingFile	This properties file must exist on the Tomcat process classpath. For example, in the ../conf/vhost-mappings.properties. The file contains mappings from the 'incoming' vhost (server name) to the vhost that should be used as the prefix in the response time log file name. If no mapping is present (no file or no entry response times are set), then the incoming vhost (server name) is used. For example:
	`pickeldi.users.acme.com=pickeldi`
	`pickeldi=`
	`%HOST%=`
	The first mapping states that if the incoming vhost is 'host1.users.acme.com', then the log file name should get a vhost of 'host1' as prefix, separated by a _ from the context root portion. The second mapping states that if the 'incoming' vhost is 'host1', then no prefix, and no _, should be used. The third mapping uses a special left-hand-side token, '%HOST%'. This mapping states that if the 'incoming' vhost is a representation of localhost then no prefix, and no _ , should be used.
	%HOST% will match the host name, or canonical host name or IP address, as returned by the implementation of InetAddress.getLocalHost().
	The second and third mappings are examples of empty right hand side, but could just as well have provided a vhost.
	This is a one time replacement. There is no recursion in the form that the result of the first line would then be applied to the second one.

15.2. Configuring HTTP Response Time Metrics

Configuring response time metrics is in some respects analogous to configuring events. The JBoss ON agent polls certain log files kept by the web server to identify the performance times for different resources served by the web server.

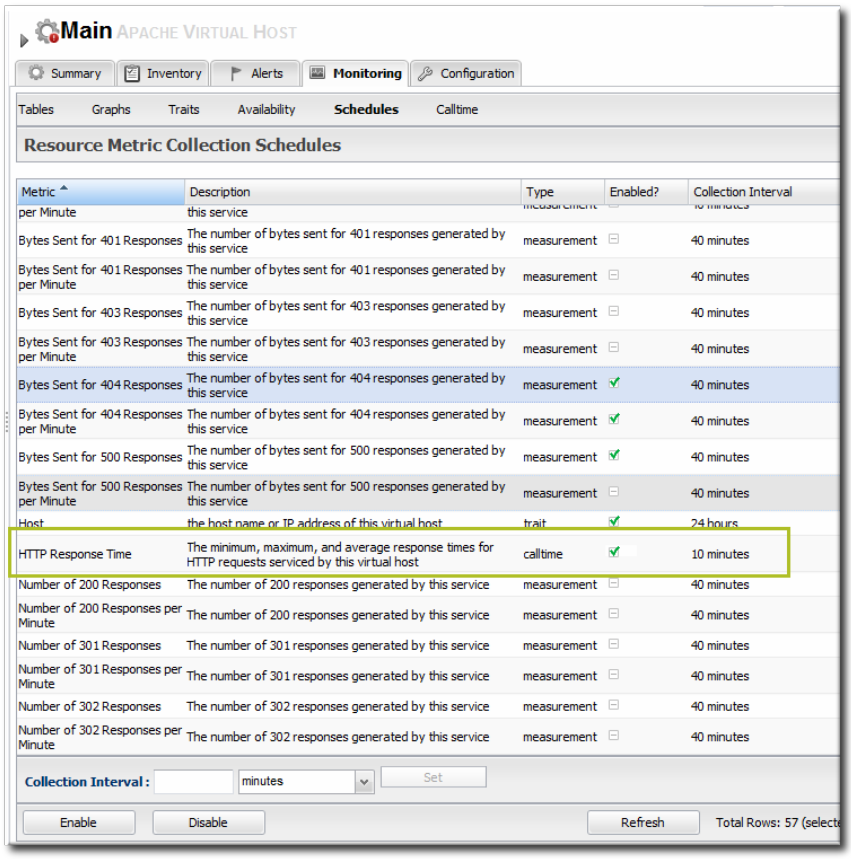
1. Install the response time filter for the Apache or Tomcat server. For Apache, simply install the filters; for Tomcat, there is additional configuration required to set up the filter entry in the **`web.xml`** file.
2. Click the **`Inventory`** tab in the top menu.
3. Select the **`Servers`** menu table on the left, and then navigate to the web server



4. Click the **`Connection Settings`** tab on the web server resource entry. and scroll to the **`Response Time`** configuration section.
5. Configure the response time properties for the web server. The agent has to know what log file the web server uses to record response time data.

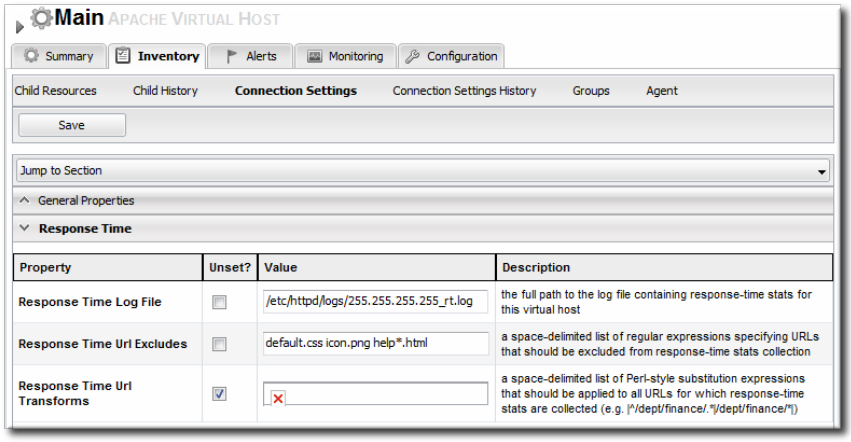
   Optionally, the server can perform certain transformations on the collected data.

   - The response times log records times for all resources the web server serves, including support files like CSS files and icons or background images. These resources can be excluded from the response time calculations in the **`Response Time Url Excludes`** field.
   - The same page can be accessed with different parameters passed along in the URL. The **`Response Time Url Transforms`** field provides a regular expression that can be used to strip or substitute the passed parameters.



6. Click the **`Save`** button.
7. Click the **`Monitoring`** tab on the web server resource entry.
8. Click the **`Schedules`** subtab.

9. Select the **`HTTP Response Time`** metric. This metric is the *calltime* type.



10. Click the **`Enable`** at the bottom of the list.

Chapter 16. Managing mod_cluster Deployments for JBoss EAP 5 (Tech Preview)

The **mod_cluster** module provides intelligent, dynamic load balancing for web applications. There are two halves to **mod_cluster**: one in the JBoss application server (which manages the web application contexts) and one in the HTTP server (which manages sessions and routing). JBoss ON monitors and manages the **mod_cluster** module within the JBoss server.

16.1. About mod_cluster

mod_cluster is an HTTP load balancer that provides a level of intelligence and control over web applications that is not available with other HTTP load balancers. **mod_cluster** has lots of features that improve performance and management, but two are crucial:

- By using multicast (*advertising*), **mod_cluster** signals workers what proxy servers are available, so workers can register themselves dynamically with the cluster domain.
- Using a special communication layer between the JBoss server and the HTTP server, **mod_cluster** can not only register when a web context is enabled but also when it is disabled and removed from load balancing. This allows **mod_cluster** to handle full web application life cycles.

More detail about the features of **mod_cluster** is in the product documentation at http://www.jboss.org/mod_cluster.

mod_cluster has two modules: one for the HTTP server which handles routing and load balancing and one for the JBoss server to manage the web application contexts. Both modules must be installed and configured for the cluster to function.

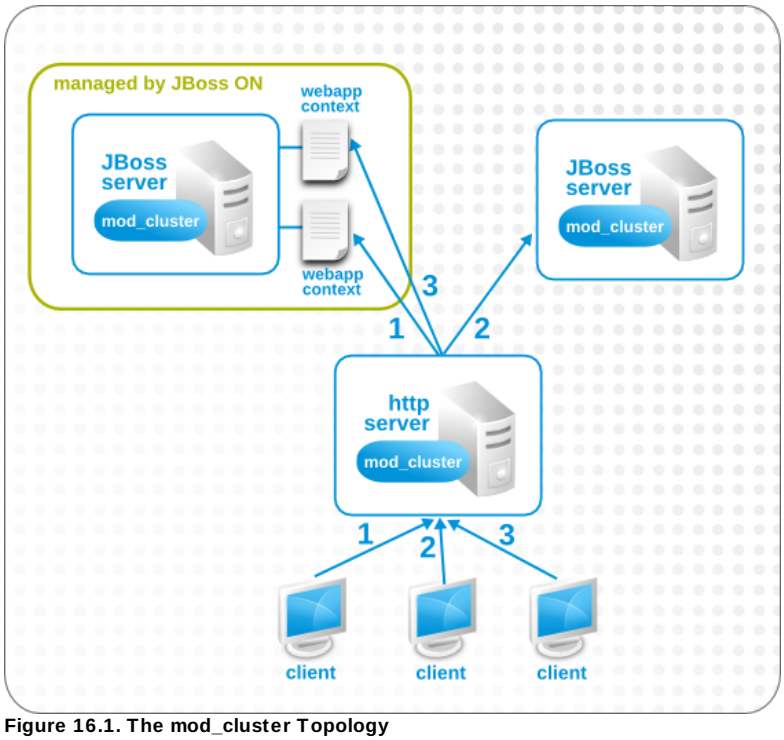


Figure 16.1. The mod_cluster Topology

In JBoss ON, the entire **mod_cluster** domain is imported as a child resource for the JBoss server. The web application contexts are listed as children resources for the cluster, with contexts as children within the **mod_cluster** module.

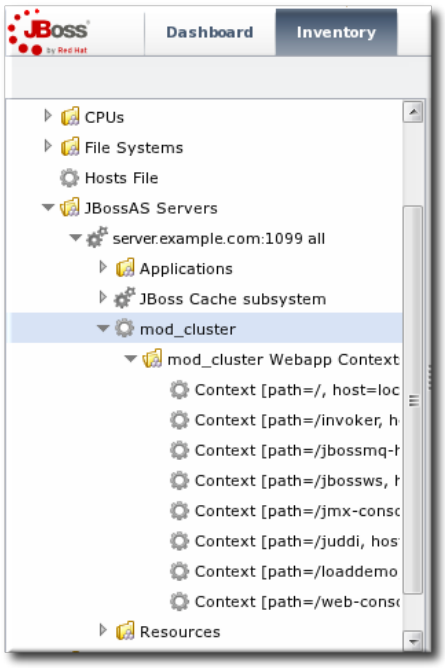


Figure 16.2. The mod_cluster Resource Hierarchy

IMPORTANT

The **mod_cluster** module in the HTTP server is configured externally from JBoss ON and is not managed by JBoss ON.
The **mod_cluster** module in the JBoss server can be managed by JBoss ON, and it is critical that the cluster is properly configured in order for JBoss ON to manage its resources. JBoss ON detects **mod_cluster** like any other JMX resource (such as Hibernate).
There are a number of resources available for installing and configuring **mod_cluster**:

- http://docs.jboss.org/mod_cluster/1.1.0/html/
- http://docs.redhat.com/docs/en-US/JBoss_Enterprise_Application_Platform/5/html/HTTP_Connectors_Load_Balancing_Guide/index.html

16.2. Managing mod_cluster

The **mod_cluster** properties provide direct management over how the **mod_cluster** domain operates. Almost any part of the **mod_cluster** configuration can be managed through JBoss ON, but two elements are critical for domain behavior:

- How the **mod_cluster** domain handles sticky sessions. Sticky sessions are enabled in **mod_cluster** by default, but this can be disabled or its behavior can be changed through the configuration properties.
- Enabling advertising (multicast). **mod_cluster** can send the JBoss information to any VirtualHost configured in the HTTP server. This allows workers to find the cluster and register themselves with the JBoss server dynamically.

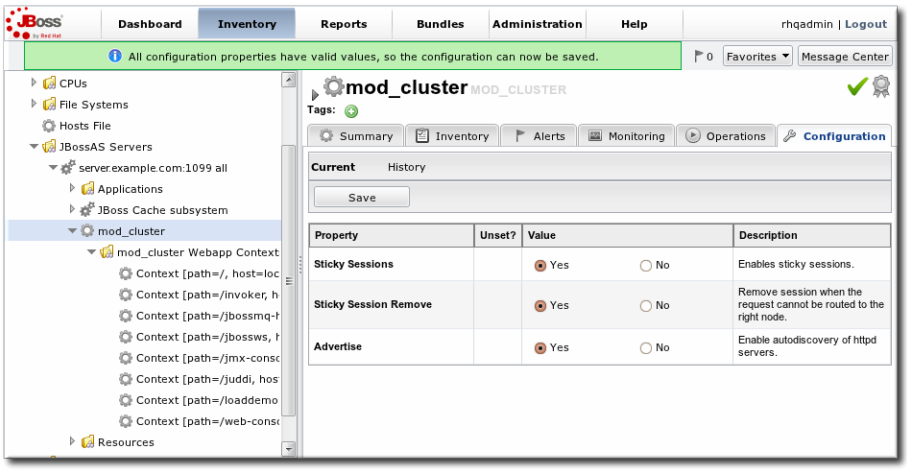


Figure 16.3. Setting Server-Level Properties

The server-level operations apply to all web application contexts configured within the **`mod_cluster`** domain. The obvious ones that impact the web application contexts are enabling and disabling all contexts. The other options are used to reset the **`mod_cluster`** domain after an error (reset the node) or reload the cluster configuration after making changes to the cluster properties.

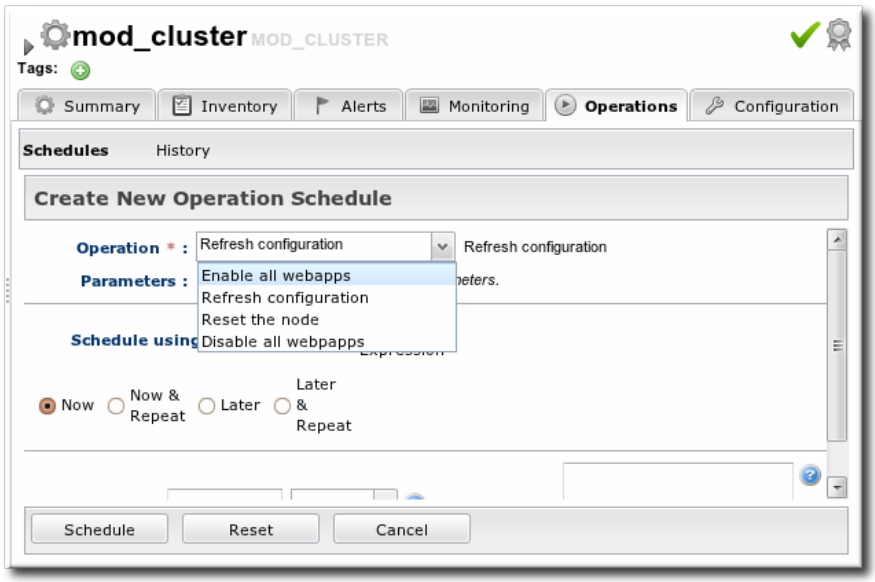


Figure 16.4. Running Server-Level Operations

16.3. Managing Web Applications Contexts

JBoss ON manages the full lifecycle of web application contexts within the **`mod_cluster`** load-balancing cluster.

Web application contexts can be stopped or disabled. Stopping or disabling a webapp context removes it from load-balancing balancing so that the Apache server cannot forward requests to the webapp, but it leaves the application running and available directly from the JBoss server address. (Stop allows the webapp context to drain before removing it from the load-balancing, so this essentially shuts down the webapp gracefully. It can take several minutes or even hours for the webapp context to stop. Disabling a webapp context immediately removes it from load balancing.)

JBoss ON has operations that allow JBoss administrators to manage the state of their web contexts within the **`mod_cluster`** domain.

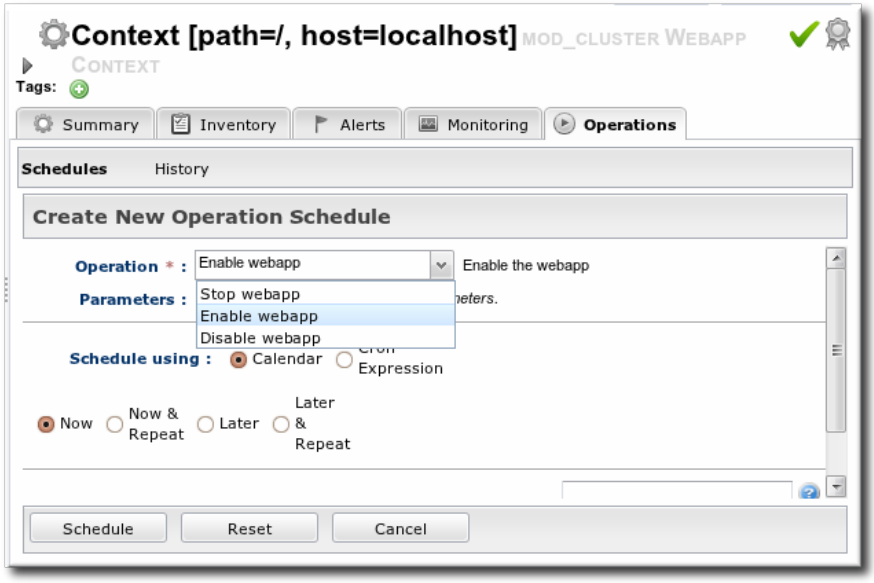


Figure 16.5. Running Web Application Context Operations

Web context resource operations only apply to the specific selected context.

Interactions with System Users for Agents and Resources

The agent runs as a specific system user, and so do servers such as JBoss and Apache which are managed by JBoss ON. The general assumption with many of the agent management tasks, including discovery, is that the agent user is the same as the resource user. If the users are different, then that can have an impact on how resources can be discovered and managed.

The common types of servers which JBoss ON manages are:

- JBoss EAP servers
- PostgreSQL databases
- Tomcat servers
- Apache servers
- Generic JVMs

For some management operations initiated by the JBoss ON agent, the agent system user is never even involved. For example, the JBoss EAP plug-in connects to the EAP instance using authentication mechanisms managed by JBoss EAP itself, so no system ACLs or user permissions are required. As long as the user can access the JBoss EAP instance, everything works.

Table A.1. Cheat Sheet for Agent and Resource Users

Resource	User Information
PostgreSQL	No effect for monitoring and discovery. The agent user must have read/write permissions to the PostgreSQL configuration file for configuration viewing and editing.
Apache	No effect for monitoring and discovery. The agent user must have read/write permissions to the Apache configuration file for configuration viewing and editing.
Tomcat	Must use the same user or can't be discovered
JMX server or JVM	Different users are fine when using JMX remoting; cannot be discovered with different users and the attach API
JBoss AS/EAP	Different users are all right, but requires read permissions on run.jar and execute and search permission on all ancestor directories for run.jar

A.1. The Agent User

There is a general assumption that the agent runs as the same user as the managed resources, and this is the cleanest option for configuration.

When the JBoss ON agent is installed from the agent installer JAR file, the system user and group who own the agent installation files is the same user who installs the JAR. So, a special system user can be created or selected, and then the agent can be installed by that user.

A.2. Agent Users and Discovery

An agent discovers a resource by searching for certain common properties, such as PIDs and processes or start scripts.

It does not necessarily matter whether the agent has superior privileges as the resource user.

For most resources, the agent simply requires read access to that resource's configuration. For resources like Apache and Postgres, as long as the agent can read the resource configuration, the resources can be discovered.

For some other resources, the agent user has to have very specific permissions:

- For JBoss EAP resources, the agent must have read permissions to the **`run.jar`** file, plus execute and search permissions for every directory in the path to the **`run.jar`** file.
- If a JVM or JMX server is running with JMX remoting, then it can be discovered if the agent is running as a different user. However, if it is running with using the attach API, it has to be running as the same user as the agent for the resource to be discovered.

A.3. Users and Management Tasks

The system user which the agent runs as impacts several common agent tasks:

- Discovery
- Deploying applications (related to Section 13.1, “Setting Permissions for Agent and Resource Users”)
- Executing scripts
- Running start, stop, and restart operations

- Creating child resources through the JBoss ON UI
- Viewing and editing resource configuration

The key thing to determine is what tasks need to be performed and who needs to perform that operation, based on limits on the resource or the operating system for permissions or authorization.

For some actions — discovery, deploying applications, or creating child resources — setting system ACLs that grant the agent user permission are sufficient.

For running operations or executing scripts, it may be necessary to run the task as a user other than the agent user. This can be done using **sudo**.

Whatever method, the goal is to grant the JBoss ON user all of the required system permissions necessary to carry out the operations.

A.4. Using sudo with JBoss ON Operations

The time to use **sudo** is for long-running operations, such as starting a service or a process, or for scripts which are owned by a resource user. The user which executes the script should be the same as the resource user because that user already has the proper authorization and permissions.

The user can really be the same, or the JBoss ON user can be granted **sudo** rights to the given command.

When elevating the agent user's permissions, two things must be true:

- There can be no required interaction from the user, including no password prompts.
- It should be possible for the agent to pass variables to the script.

To set up **sudo** for resource scripts:

1. Grant the JBoss ON agent user **sudo** rights to the specific script or command. For example, to run a script as the **jbossadmin** user:

   ```
   [root@server ~]# visudo

   jbosson-agent     hostname=(jbossadmin)  NOPASSWD: /opt/jboss-eap/jboss-
   as/bin/*myScript*.sh
   ```

   Using the **NOPASSWD** option runs the command without prompting for a password.

   IMPORTANT

   JBoss ON passes command-line arguments with the start script when it starts an EAP instance. This can be done either by including the full command-line script (including arguments) in the **sudoers** entry or by using the **sudo -u** *user* command in a wrapper script or a script prefix.
   The second option has a simpler **sudoers** entry

2. Create or edit a wrapper script to use. Instead of invoking the resource's script directly, invoke the wrapper script which uses **sudo** to run the script.

   TIP

   For the EAP start script, it is possible to set a script prefix in the connection settings, instead of creating a separate wrapper script:

   ```
   /usr/bin/sudo -u jbosson-agent
   ```

   For example, for a start script wrapper, **start-myScript.sh**:

   ```
   #!/bin/sh
   # start-myScript.sh
   # Helper script to execute start-myConfig.sh as the user jbosson-agent
   #
   sudo -u jbosson-agent /opt/jboss-eap/jboss-as/bin/start-myConfig.sh
   ```

3. Create the start script, with any arguments or settings to pass with the **run.sh** script. For example, for **start-myConfig.sh**:

   ```
   nohup ./run.sh -c MyConfig -b jonagent-host 2>&1> jboss-MyConfig.out &
   ```

Index

D

deploying apps

- timeout, Deploying Web Applications to a Domain, Deploying a Web Application as a Child Resource
- troubleshooting, Deploying Web Applications to a Domain, Deploying a Web Application as a Child Resource

discovery
- secured JMX server, Enabling the Agent to Connect to Secured JMX Servers

J

JBoss
- applying patches, Applying JBoss Patches from the Patch RSS Feed
- enabling the default patch content source, Enabling the Default JBoss Patch Content Source
- managing and configuring, How JBoss ON Manages JBoss Resources
- subscribing resources to the default patch, Subscribing Multiple JBoss Resources to the Default JBoss Patch Repository

JBoss EAP
- discovering secure JMX servers, Enabling the Agent to Connect to Secured JMX Servers

JBoss EAP 6/AS 7
- configuring response time metrics, Configuring Response Time Metrics for JBoss EAP 6/AS 7

JBoss EAP/AS 5
- configuring response time metrics, Configuring HTTP Response Time Metrics for JBoss EAP/AS 5

JMX server
- discovery, Setting up a Custom JVM for Discovery, Enabling the Agent to Connect to Secured JMX Servers
  - excluding processes, Excluding Java Processes from Discovery
  - required configuration, Required JVM Configuration for Discovery
- manually importing, Manually Importing a JVM Resource

JVM
- discovery, Setting up a Custom JVM for Discovery
  - excluding processes, Excluding Java Processes from Discovery
  - required configuration, Required JVM Configuration for Discovery
- manually importing, Manually Importing a JVM Resource

M

monitoring
- configuring response time filters for JBoss EAP 6/AS 7 servers, Configuring Response Time Metrics for JBoss EAP 6/AS 7
- configuring response time filters for JBoss EAP/AS 5 servers, Configuring HTTP Response Time Metrics for JBoss EAP/AS 5

P

patches
- applying, Applying a Patch

R

resources
- child
  - connection factories, Creating Connection Factories
  - data sources, Creating Data Sources
  - EAR and WAR, Deploying EAR and WAR Files
  - JMS queues and topics, Creating JMS Queues and Topics

- child types, Deploying Applications

response time filters
- configuring JBoss EAP 6/AS 7, Configuring Response Time Metrics for JBoss EAP 6/AS 7
- configuring JBoss EAP/AS 5, Configuring HTTP Response Time Metrics for JBoss EAP/AS 5

T

timeout
- manually discover app, Deploying Web Applications to a Domain, Deploying a Web Application as a Child Resource